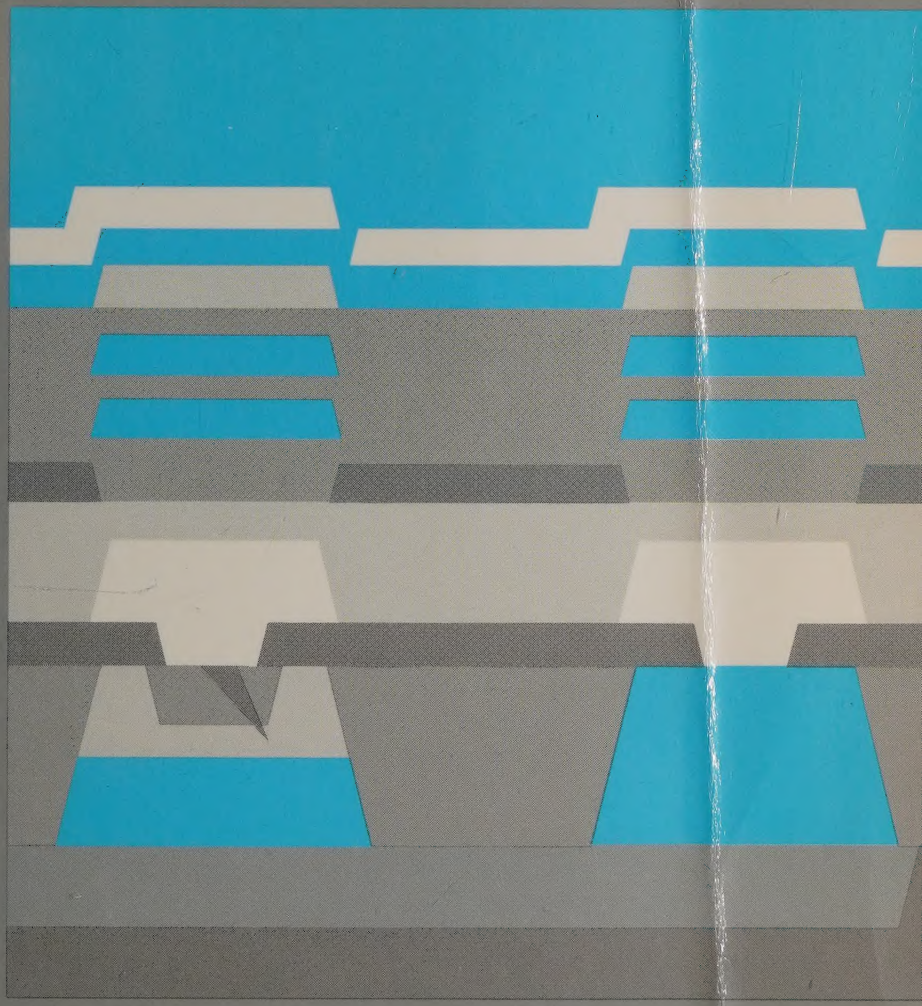
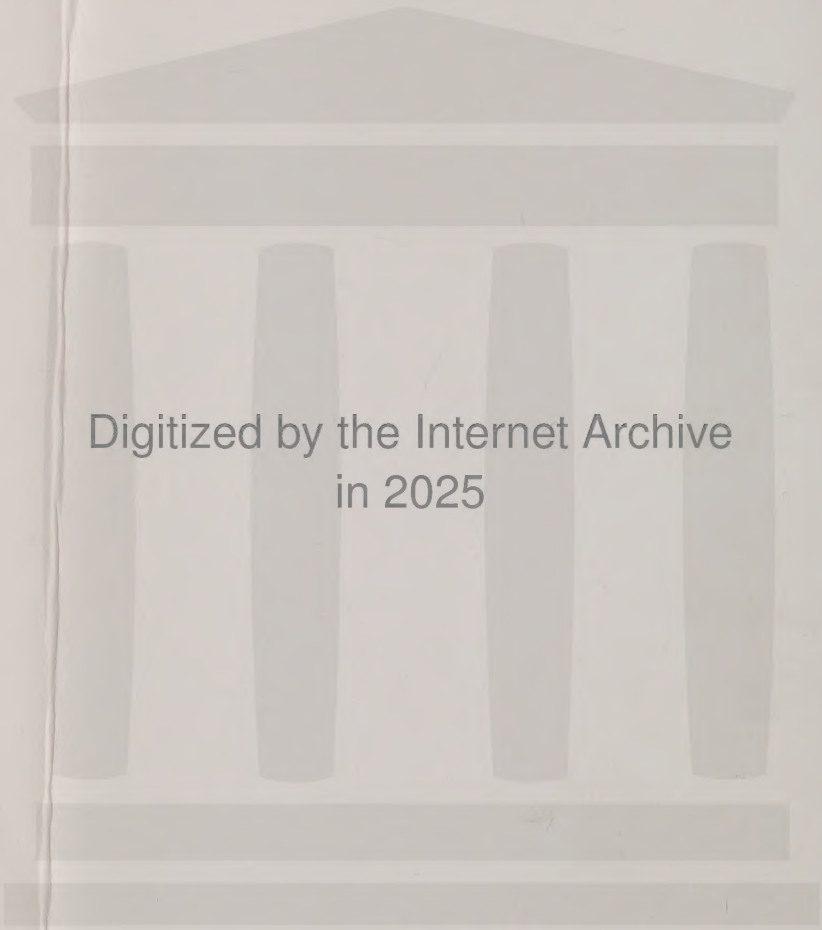
November, 1985

HITACHI

HD64180 ASE-I USER'S MANUAL



#U78



Digitized by the Internet Archive
in 2025

https://archive.org/details/manualsplus_07452

HD64180 ASE-I
USER'S MANUAL

HITACHI

WARNING

This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

When using this manual, the reader should keep the following in mind:

1. This manual may, wholly or partially, be subject to change without notice.
2. All rights reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this manual without Hitachi's permission.
3. Hitachi will not be responsible for any damage to the user that may result from accidents or any other reasons during operation of his unit according to this manual.
4. This manual neither ensures the enforcement of any industrial properties or other rights, nor sanctions the enforcement right thereof.
5. Circuitry and other examples described herein are meant merely to indicate characteristics and performance of Hitachi semiconductor-applied products. Hitachi assumes no responsibility for any patent infringements or other problems resulting from applications based on the examples described herein.

November 1985
©Copyright 1985, Hitachi America Ltd.
Printed in U.S.A.

PREFACE

This manual provides a general description of the HD64180 Adaptive System Evaluator (ASE) H180AS01 and shows how to use it.

The ASE adequately supports the development of the user system using Hitachi 8-bit microcomputer for software and hardware.

The ASE can operate by itself with a console, but provides more powerful support with a development system such as the IBM-PC, DEC VAX, or the Hitachi H680SD200 (hereinafter abbreviated as SD200).*

Please refer to "11. PRECAUTIONS" before using this system.

*HITACHI SD200 IS NOT AVAILABLE IN THE U.S.A.

CONTENTS

	Page
1. OVERVIEW	1
1.1 Features	2
1.2 ASE Configuration	3
1.2.1 Hardware	3
1.2.2 Software	4
1.3 Functional Description	5
2. SPECIFICATIONS	9
2.1 System Components	9
2.2 Environmental Conditions	11
2.3 ASE Station Panel	12
2.3.1 Front panel	12
2.3.2 Rear panel	13
2.4 Buffer Box	14
3. INSTALLATION HARDWARE	15
3.1 Connecting the Buffer Box	15
3.2 User System Interface Cable	17
3.3 Installating the Emulation Memory Board	18
3.4 System GND Connections	20
4. SYSTEM CONFIGURATION	22
4.1 Connecting with SD200	24
4.1.1 Connecting procedure with SD200	25
4.2 Connecting with Console	26
4.2.1 Console interface	26
4.2.2 Selection of RS-232C baud rate	29
4.2.3 Break key	31
4.3 Connecting with General Purpose Host System	32
4.3.1 Specification of host interface hardware	33
4.3.2 Protocol	36
4.4 Connecting with Printer	44
5. ASE FUNCTIONS AND INTERFACING WITH USER SYSTEM	52
5.1 Real-time Emulation	52
5.1.1 Trigger mode	52
5.1.2 Cycle reset mode	53
5.2 Break	53
5.3 Trace	54
5.4 Single Step	57
5.5 Measurement of Execution Time	57
5.6 Memory Access	57
5.7 Monitoring Emulation	57
5.8 Control and Condition Check of the User MPU	59
5.9 Coverage	61
5.10 Software Development Support and Data Management	61
5.11 User System Interface Circuit	63

Page

6. HD64180 I/O FUNCTION SUPPORT ---------- 71
6.1 I/O Space Access ---------- 71
6.2 Logical Address and Physical Address (MMU function) ---------- 71
6.2.1 Address display at ASE command execution ---------- 72
6.2.2 Logical address in physical address input mode ---------- 72
6.2.3 Physical address in logical address input mode ---------- 75
6.2.4 Address range input by logical address ---------- 76
6.2.5 Logical address in TRACE_BREAK_CONDITION and STOP commands ----- 79
6.3 DMA Function ---------- 79
6.4 Refresh Function ---------- 80
6.5 Timer Function ---------- 80
6.6 ASCI Function and CSI/O Function ---------- 80
6.7 Forced Input Reset Signal from ASE ---------- 81

7. ASE STARTING UP ---------- 82
7.1 Starting Procedure ---------- 82

8. COMMAND INPUT AND DISPLAY ---------- 85
8.1 Command Syntax ---------- 85
8.2 How to Input One-line ASSEMBLE Command ---------- 86
8.3 Special Key Function ---------- 87
8.4 Full Screen Display Mode ---------- 89

9. ASE COMMANDS ---------- 92
9.1 ADDRESS ---------- 93
9.2 ADDRESS_MODE ---------- 95
9.3 ASSEMBLE ---------- 97
9.4 BACKUP ---------- 100
9.5 BASE ---------- 102
9.6 BREAK ---------- 106
9.7 BREAK_MASK ---------- 112
9.8 CHECK ---------- 115
9.9 CLOCK ---------- 118
9.10 COMMAND_CHAIN ---------- 120
9.11 CONVERT ---------- 121
9.12 COVERAGE ---------- 123
9.13 DISASSEMBLE ---------- 128
9.14 DISPLAY ---------- 130
9.15 FLOPPY_FORMAT ---------- 134
9.16 FLOPPY_LOAD ---------- 135
9.17 FLOPPY_SAVE ---------- 136
9.18 FLOPPY_VERIFY ---------- 138
9.19 GO ---------- 140
9.20 HELP ---------- 148

Page

9.21 HOST ---------- 149
9.22 IO_CONTROL ---------- 152
9.23 LOAD ---------- 155
9.24 MAP ---------- 158
9.25 MEMORY ---------- 161
9.26 MOVE ---------- 166
9.27 MOVE_TO_RAM ---------- 167
9.28 PRINT ---------- 169
9.29 QUIT ---------- 171
9.30 REGISTER ---------- 172
9.31 RESULT ---------- 173
9.32 SAVE ---------- 174
9.33 STATUS ---------- 177
9.34 STEP ---------- 178
9.35 STOP ---------- 182
9.36 SYMBOL ---------- 186
9.37 TRACE ---------- 189
9.38 TRACE_BREAK_CONDITION ---------- 195
9.39 TRACE_POINTER ---------- 203
9.40 VERIFY ---------- 206
9.41 .<register name> ---------- 210

10. ERROR MESSAGES ---------- 213

11. PRECAUTIONS ---------- 218
11.1 Precautions before System Start-up ---------- 218
11.2 Precautions at Emulation ---------- 219

12. MAINTENANCE ---------- 223
12.1 Self Diagnostics ---------- 223
12.2 Troubleshooting ---------- 224

Appendix A. The ASE Load Module Format ---------- 227
A.1 Motorola S-type Format ---------- 228
A.2 INTELLEC HEX Type Format ---------- 229
B. ASCII Code ---------- 230
C. Hexadecimal-Decimal Number Conversion Table ---------- 231
D. 2's Exponents ---------- 236

HITACHI SALES OFFICE LOCATIONS ---------- 237

This system is an adaptive system evaluator to support the development of an applied system using an 8-bit microcomputer HD64180.

The HD64180 provides a high-speed CPU, Memory Management Unit (MMU), Direct Memory Access Controller (DMAC), Programmable Reload Timer (PRT), Asynchronous Serial Communication Interface (ASCI), Clocked Serial I/O Port (CSI/O) on single chip. Because of these benefits, the HD64180 is useful in system applications requiring high performance. Therefore, system development requires an excellent debugging and evaluation tool for effective debugging.

The ASE enables real-time emulation and provides a powerful function to improve the hardware/software of the user system for debug-efficiency. Fig. 1-1 shows the ASE appearance.

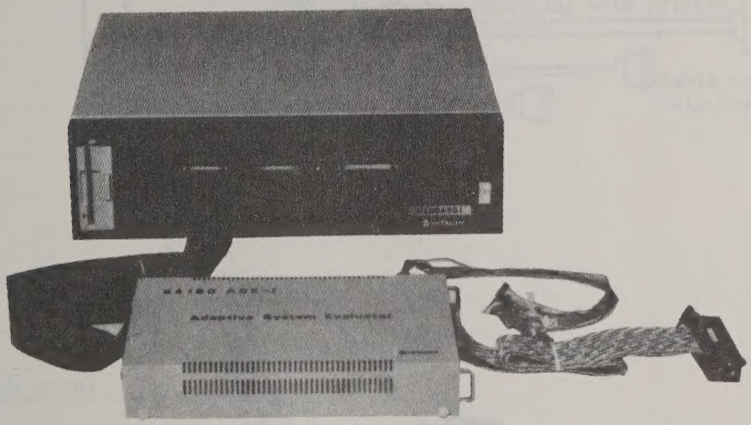


Fig. 1-1 H180AS01 Appearance

1.1 Features

The features of the ASE are as follows.

(1) Provides real-time emulation (0.5 - 6.144 MHz).
(2) Provides plentiful emulation commands for efficient development.
(3) Used as a stand-alone system with a console.
    . RS-232C interface
(4) Includes a 3.5-inch micro floppy disk drive as standard equipment.
    . Loads, saves, and verifies the memory contents in the user system
(5) Offers more powerful debugging by connecting to the SD200.
(6) Can load, save and verify the object program by connecting to the host system except SD200.
    . RS-232C interface

(7) Can be used to connect to a printer.
    . Centronics interface

1.2 ASE Configuration

1.2.1 Hardware

The ASE consists of ASE station, buffer box, MPU connector and eight external probes as shown in Fig. 1-2. The ASE station is connected to the host development system SD200, other host development systems, or console (CRT) with two RS-232Cs.

Also, the ASE is connectable with a printer (Centronics).

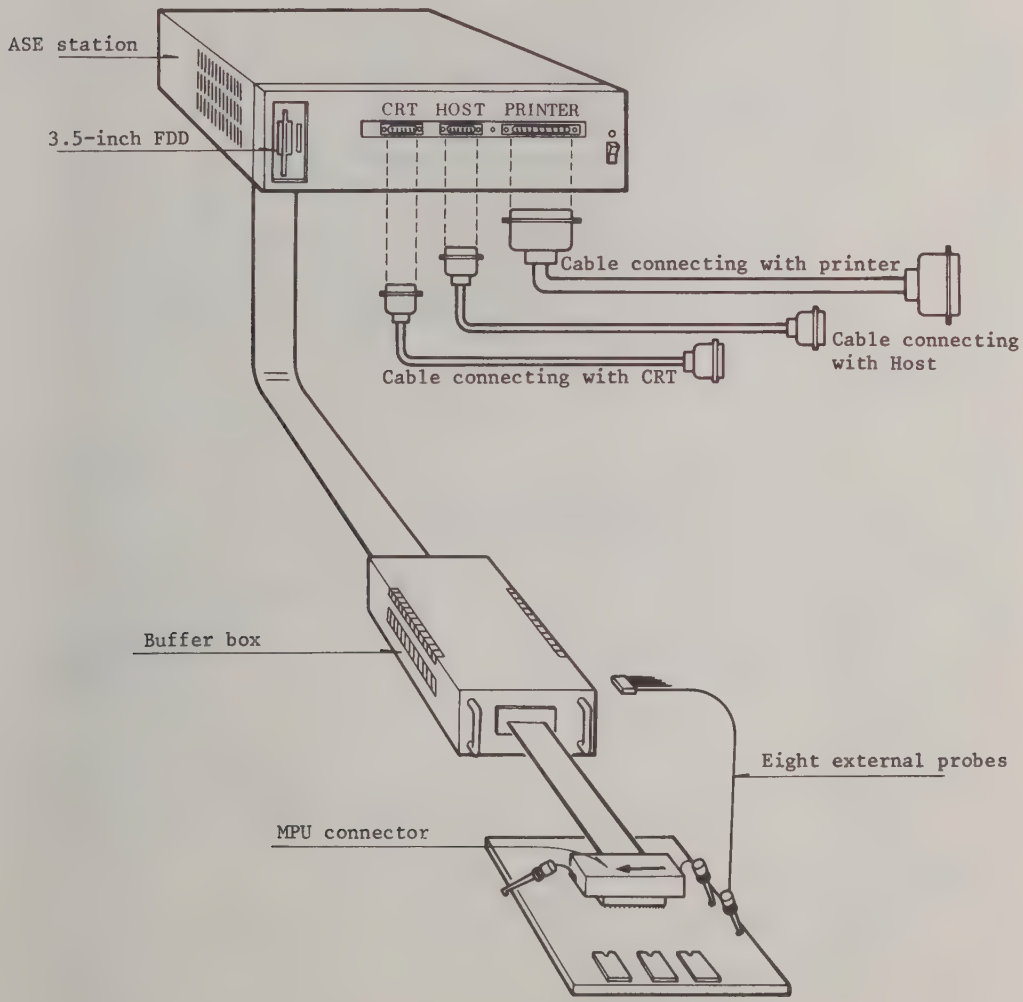


Fig. 1-2 ASE Hardware Configuration

1.2.2 Software

The ASE software includes a monitor (EPROM) controlling I/O and system program (3.5-inch floppy disk) controlling the buffer box in the ASE station, and also includes user system control software (EPROM) in the buffer box controlling the data communication between the ASE station and the user system.

When connecting to the SD200 using the SD200 interface program, the CRT editor and cross assembler are available and effective debugging is possible.

Fig. 1-3 shows the software configuration when connected with SD200.

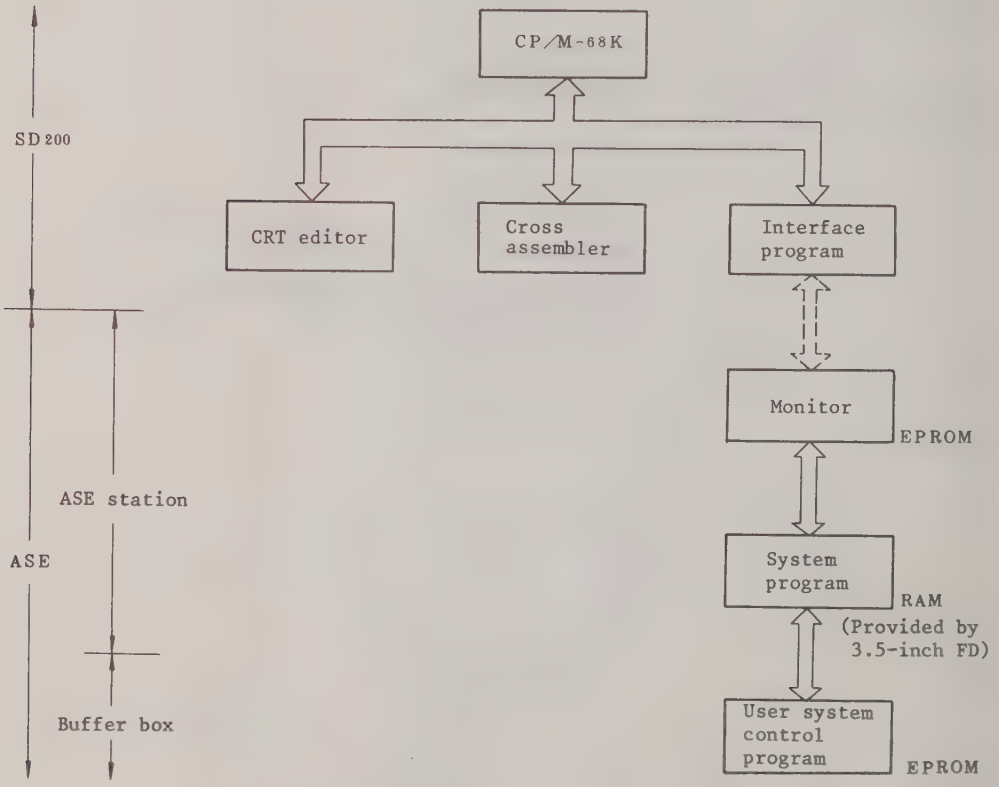


Fig. 1-3 Software Configuration When Connected with SD200

1.3 Functional Description

Table 1-1 provides the functional description of the ASE.

Table 1-1 Functional Description

No.	Item	Description	Remarks
1	Real-time Emulation	(1) Real-time emulation (4 to 6.144 MHz) (0.5 to 6.144 MHz with external clock) . Emulation until the software break conditions and hardware break conditions are satisfied, and the break key is pressed.	Refer to "9.19 GO"
		(2) Emulation with the trigger output as a synchronous signal for an oscilloscope under hardware break conditions. · Pulse output under hardware break conditions when using TRIG output pin on the ASE station front panel as a synchronous signal for an osilloscope. (Low Pulse)	
		(3) Emulation with input of the RESET into MPU every specified time (cycle reset mode) · Uses the cycle reset mode for debugging the waveform observation from RESET input to the specified time.	
2	Break Condition Setting	(1) Hardware break condition setting · The following conditions specify two break-points to force an emulation stop. · Program counter · Address bus, data bus · External probes · Range of program counter or address · Number of condition occurrence	"9.38 TRACE_BREAK_CONDITION"
		(2) Software break condition setting · Sets breakpoints in the program of RAM area (255 breakpoints max.)	Refer to "9.6 BREAK".
3	Trace Display with Real-time Emulation	(1) Displays executed instruction in mnemonics	Refer to "9.37 TRACE".
		(2) Displays the executed bus cycle or the following information every 125 nS. · Mnemonic instruction · Address bus, data bus · External signals(eight probes) · I/O signals of MPU	Refer to "9.19 GO", and "9.37 TRACE".
		(3) Trace condition setting · Trace start condition setting · Trace save condition setting	Refer to "9.38 TRACE_BREAK_CONDITION ".

4	Single-step Emulation	(1) Emulation with a display of the following information every single step • Mnemonic instruction • Contents of registers	Refer to "9.34 STEP".
		(2) Emulation with a display of the information mentioned in (1) only for the branch instruction.	Refer to "9.34 STEP".
		(3) Stop condition setting • Program counter • Address bus, Data bus • Content of register • Contents of memory	Refer to "9.35 STOP".
5	Memory Access	(1) Display and modification of contents of memory • Display and modification of contents of memory space and I/O space.	Refer to "9.25 MEMORY".
		(2) Specifications of memory attributes in 4KB (only for memory space) • Memory on the user system • Emulation memory • Write inhibition specification	Refer to "9.24 MAP".
6	Emulation Support	(1) Emulation memory(Standard optional 8KB) • 256KB memory board (Max. two boards) (Optional) (H680SM01S)	Refer to "9.24 MAP".
		(2) MPU clock • 6.144 MHz • User system clock	Refer to "9.9 CLOCK".
7	Line Assemble	(1) Assemble of mnemonic instruction and memory setting	Refer to "9.3 ASSEMBLE".
8	Dis-assemble	(1) Disassembles contents of memory at the specified address and displays the mnemonic instruction.	Refer to "9.13 DIS-ASSEMBLE".
9	Load, Save, and Verify the User Program	(1) Loads, saves, and verifies the user program with 3.5-inch floppy disk.	Refer to "9.16 FLOPPY_LOAD", "9.17 FLOPPY_SAVE", and "9.18 FLOPPY_VERIFY".

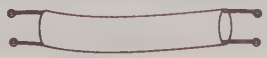
		(2) Loads, saves, and verifies the user program from the file in the host system.	Refer to "9.23 LOAD", "9.32 SAVE", and "9.40 VERIFY".
10	MPU Signal Control and State Checks	(1) Enable/Disable function of external interrupt signals • $\overline{\text{NMI}}$ • $\overline{\text{INT0}}$ – $\overline{\text{INT2}}$	Refer to "9.22 IO_CONTROL".
		(2) Checks the MPU I/O signal • Chesks malfunction such as a short-circuit for each MPU signal.	Refer to "9.8 CHECK".
11	Execution time measurement	(1) Execution time measurement (Possible measuring time = 1 to 2^{45} μS) • After user program execution with Go command measures the execution time until the end.	Refer to "9.19 GO".
12	Coverage Trace	(1) Traces the address of the MPU accessed during user program execution	Refer to "9.12 COVERAGE".
13	Debugging by Symbol	(1) Debugs while referring to the registered symbol under the following conditions • Symbols assembled on the SD200 (but symbol including characters other than alphanumerics cannot be used.) • Symbols entered from console	Refer to "9.36 SYMBOL".
14	Others	(1) Catalogue function of ASE commands • Inputs the ASE command sequentially from the SD200 file and executes them.	Refer to "9.10 COMMAND_CHAIN" only with SD200.
		(2) Figure conversion • Logical address, physical address conversion. • Binary, octal, decimal, hexadecimal conversion, ASCII display	Refer to "9.1 ADDRESS" and "9.11 CONVERT"
		(3) Programmable function key • Sets commands corresponding to the function keys	Refer to "8.3 Special Key Function" only with SD200.
		(4) Screen display function • Displays the emulation execution.	Refer to "8.4 Full Screen Display Mode" and "9.14 DISPLAY" only with SD200.

		(5) Operation state display	Refer to "9.33 STATUS".
		(6) Monitoring emulation • After starting emulation, checks the emulation every 300 mS and displays it on the screen if there's something wrong.	Refer to "5.7 Monitoring Emulation"
		(7) Starts and stops the printer.	Refer to "9.28 PRINT"
		(8) Selects the logical or physical input mode of the address input.	Refer to "9.2 ADDRESS_MODE".
		(9) Selects the logical or physical input mode of the base type for address calculation.	Refer to "9.5 BASE".
		(10)Transfers to the emulation memory from the user system memory.	Refer to "9.27 MOVE_TO_RAM".

2.1 System Components

Please identify the ASE system components shown in Table 2-1.

Table 2-1 System Components

Category	No.	Item	Configuration	Qty.	Remarks
Hardware	1	ASE station		1	Power supply, 2 Printed circuit boards
	2	Buffer box		1	3 Flat cables, 3 Inserted PCBs
	3	RS-232C interface cable		2	For host system --- 1 (10m), For CRT---1(10m)
	4	AC cable		1	
	5	External probe		1	Probe for signal input --- 8 for GND -- 1
	6	GND cable		2	
	7	Printer interface cable		1	For printer connection --- 1 (2 m)
	8	Fuse	(3 A)	1	Spare part
	9	User system interface cable		1	Protection socket (IC83-64075GG4 manufactured by Yamaichi Electronics, Mfg. Ltd)
	10	Jacket		1	
Software	11	Floppy disk	(3.5 inch)	1	Test and maintenance program (S180TM01F)
			(3.5 inch)	1	System program (S180ASE1F)
Manual	12	64180ASE -I User's Manual		1	

A memory board is provided as optional emulation memory in addition to components shown in Table 2-1.

The option component specification is as follows;

Table 2-2 Option Component Specification

No.	Item	Specification	Function
1	Emulation memory (H680SM01S)	Used SRAM and with parity check function. One memory board (256KB) Two memory boards can be mounted.	Mounting this board, it can be used as user system memory.

Fig. 2-1 shows ASE dimensional outline and weight.

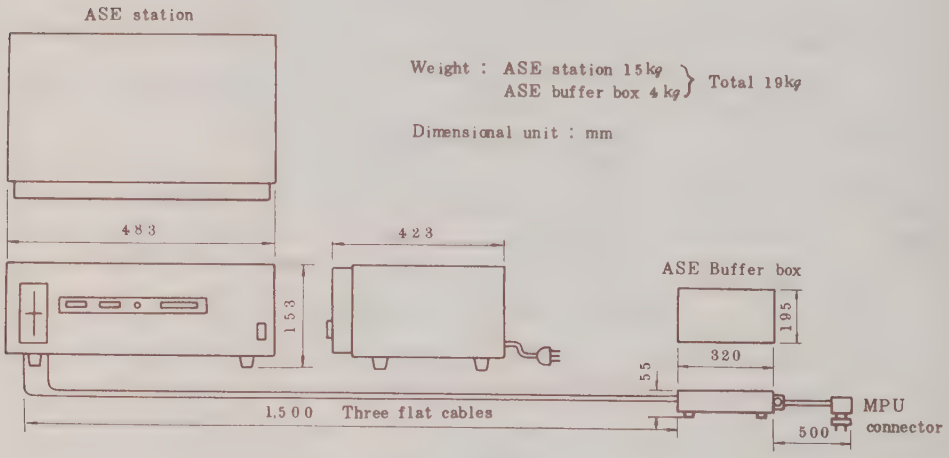


Fig. 2-1 ASE Dimensional Outline and Weight

* When connecting the ASE to SD200, the following software will be provided.
  (1) Media -- 8-inch floppy disk (SD200)
  (2) Type name -- S180CAS1F
  (3) Purpose -- Interface program between ASE and SD200

When using the ASE, observe the following conditions.

Table 2-3 Environmental Conditions for Use

No.	Item	Specifications
1	Temperature	Operating : 10 to 35°C Storage : -10 to 50°C
2	Humidity	Operating : 35 to 80 % RH (No condensation) Storage : 35 to 80 % RH (No condensation)
3	Vibration	Operating : 0.25 G max. Storage : 0.5 G max. Packed : 1.5 G max.
4	AC input power supply	Voltage : AC100 to 120V ± 10 % or 200 to 240V ± 10 % Frequency : 50/60 Hz Power consumption : 300 VA or less
5	Ambient	No corrosive gas.

Table 2-4 indicates specification of ASE built-in floppy disk drive.

Table 2-4 Specification of ASE Built-in Floppy Disk Drive

Item	Specifications
Memory capacity	Approximately 655K bytes (256x16x160) (Double sided double density track)
Recording system	MFM
Recording format	IBM format (256 bytes/sector, 16 sectors/track)
Recommended record media	MS2-DD (Hitachi Maxell)

2.3.1 Front panel

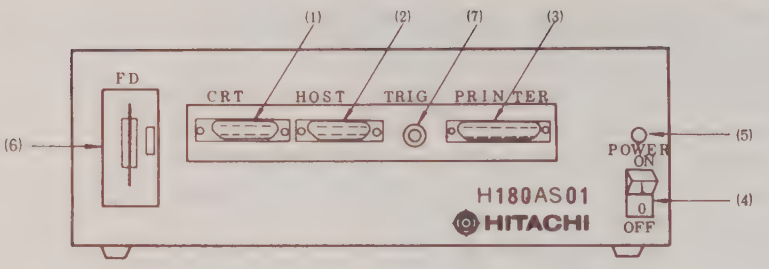


Fig. 2-2 ASE Station Front Panel

(1) Console interface connector :
Connector for console interface marked "CRT".

(2) Host system interface connector :
Connector for host development system (SD200 or general purpose host system) interface marked "HOST".

(3) Printer interface connector :
Connector for printer interface marked "PRINTER".

(4) Power switch :
The power is supplied to the ASE with the switch ON.

(5) Power lamp :
The lamp lights with the ASE power ON.

(6) 3.5-inch floppy disk drive:
The ASE system program is loaded and the memory contents in the user system is loaded, saved and verified with 3.5-inch floppy disk in the drive.

(7) TRIG output pin :
The pin outputs "Low" level pulse in the following situations, and marked "TRIG";
- When specifying a trigger mode with GO command, and the hardware break conditions are met.
- When specifying a cycle reset mode with Go command, and the RESET is input to MPU.

2.3.2 Rear panel

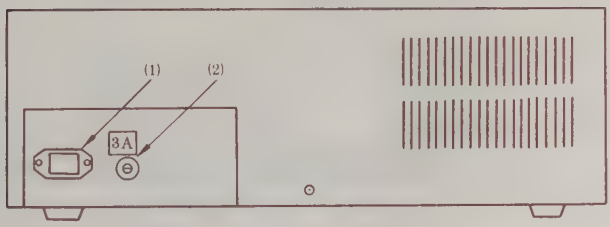


Fig. 2-3 ASE Station Rear Panel

(1) AC Power connector : Connector for AC input.

(2) Fuse Box : AC power fuse (3A) included.

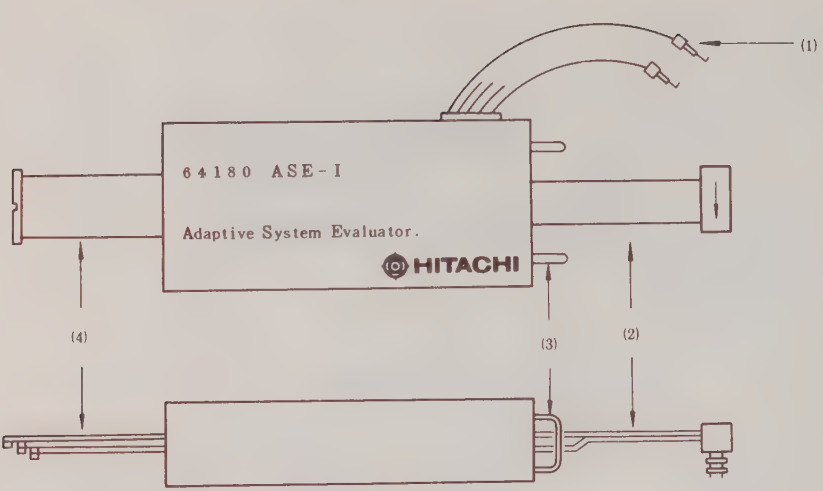


Fig. 2-4 Buffer Box

(1) External probe
The probe functions under the following conditions in user system emulation.
- Hardware break condition
- Real-time trace input

(2) User system interface cable
The user system interface cable is connected to the IC socket for HD64180 MPU in the user system and enables the HD64180 in the user system.
Note that the protection socket is connected to the MPU socket. Please use the same IC socket in the user system.
IC83-64075GG4 (Manufactured by Yamaichi Electronics Mfg. Ltd.) or its equivalent.

(3) Fixing handle :
The handle is used for fixing the buffer box to the user system if the user cannot connect the buffer box to the user system with difficulty.

(4) Flat cables :
Three flat cables connect the buffer box to the ASE station.

3.1 Connecting the Buffer Box

The ASE is shipped from the factory with its buffer box detached. Therefore, the buffer box must be installed as below. No problem is involved with the buffer box connected to the ASE station even where the ASE is not in use. If it is necessary to remove the buffer box in order to move the ASE disconnect and re-connect the buffer box as required.

(1) See that the ASE power is off. (The POWER lamp on the right side of the ASE station front panel should stay off.) In addition, both the host system and console power switches must be off.
(2) Disconnect the ASE station power cable from the outlet.
(3) Disconnect the interface cables (HOST, CRT and PRINTER) from the connectors on the ASE station front panel.
(4) Draw the ASE station front panel toward you.

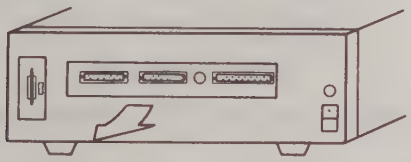


Fig. 3-1 Removing Front Panel

Inside the ASE station, the boards are installed as shown in Fig. 3-2. (Emulation memory board is optionally used.)

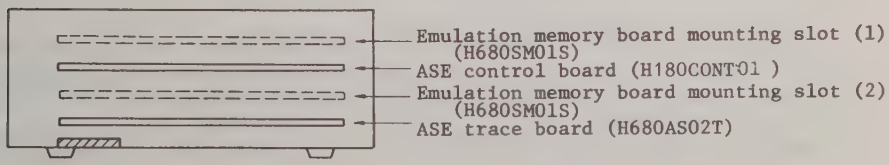


Fig. 3-2 Boards Installed in ASE Station

(5) Insert the folded part of the cable into the case at the bottom of the ASE station. (See Fig. 3-3.) The cable top must be located just under the connector of the trace board. Cable clamp must be detached.

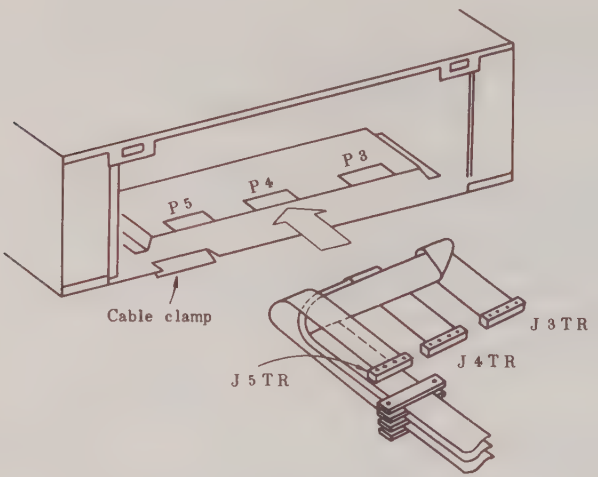


Fig. 3-3 Inserting the Cable

(6) Secure the cable clamp at the left bottom of the ASE station front. (See Fig. 3-4.)

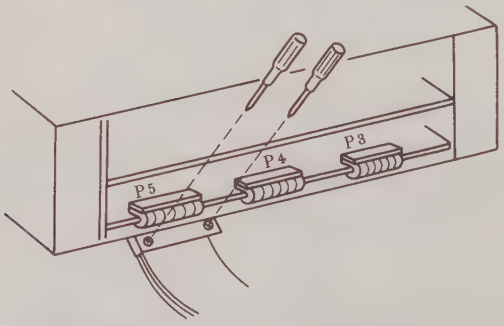


Fig. 3-4 Securing the Cable Clamp

(7) Connect cables J3TR, J4TR, J5TR, to connectors P3, P4, P5, respectively, on the trace board.
(8) Recheck the connector and cable connections. Also check the board secured by the connector, because the board might get out of connction during delivery.
(9) Replace the front panel.
(10) Connect the HOST, CRT, and PRINTER interface cables.

3.2 User System Interface Cable

The following is information about connection and the User system interface cable between the buffer box and the user system.

See Fig. 3-5 for the connection.

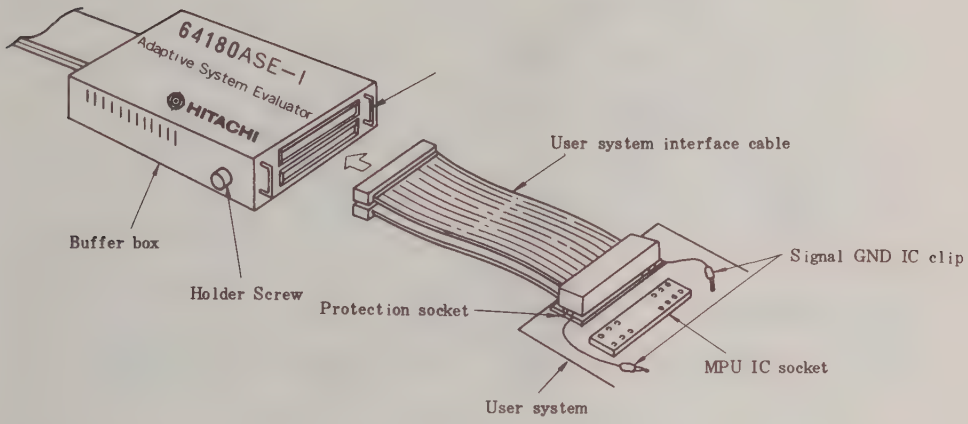


Fig. 3-5 Connecting User System Interface Cable

Before connecting the user system interface cable to the user system, check the matching of the user system with the MPU IC socket pin specifications. Please use the IC83-64075GG4 (manufactured by Yamaichi Electric Mfg. Ltd.) or its equivalent in the MPU IC socket in the user system.

There are two GND IC clips, both of which must be connected to the GND pins of the user system. These clips are intended to provide GND reinforcement; if they are not used, a malfunction may occur.

A GND cable is available, as an option, to ensure the GND reinforcement between the ASE and the user system. A more reliable system evaluation can be achieved by connecting the buffer box fixing handle and the user system frame GND.

The user system interface cable can be disconnected by reversing the connection steps. Note that when disconnecting the user system interface cable from the MPU IC socket, apply a screwdriver to either side of the user system to ease out the pins.

A jacket is also available to protect the user system interface cable from external noise such as static electricity. Use the jacket (to cover the user system interface cables) if there is a possibility of external noise causing a malfunction. Connect the GND terminal of the jacket to the GND terminal on the user system and to the holder screws on the buffer box. (See Fig. 3-6.)

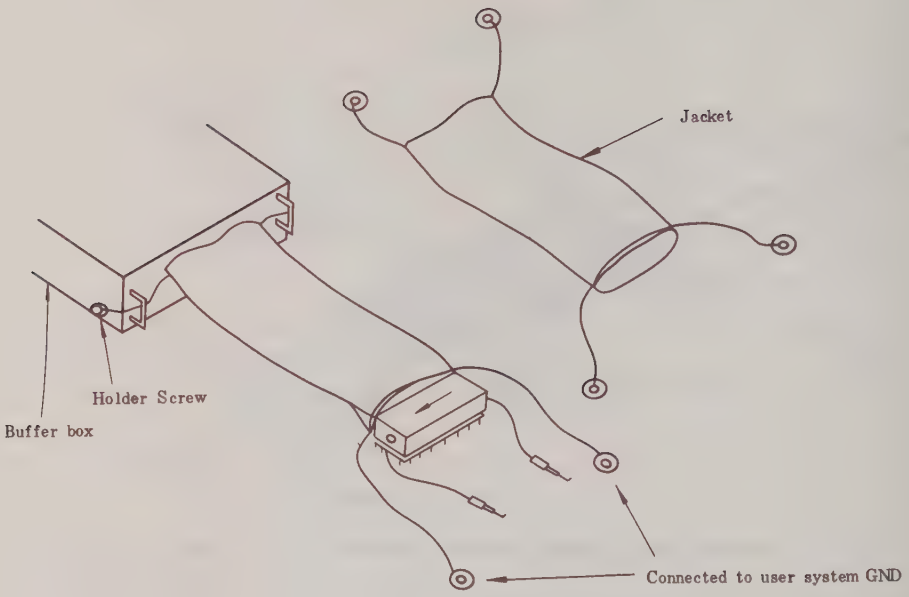


Fig. 3-6 Installing the Jacket

3.3 Installating the Emulation Memory Board

The ASE system provides a static memory board (H680SM01S) as an emulation memory for a user system (Optional).

The ASE is available with up to two emulation memory boards (256 KB per board).

Different memory address settings apply between the two emulation memory boards, with the address setting switch as shown in Fig. 3-7. Address setting on the first memory board is different from that on the second one.

The address has nothing to do with the address used as a memory in the user system. As for the unused position of the address setting switch, the setting does not matter.

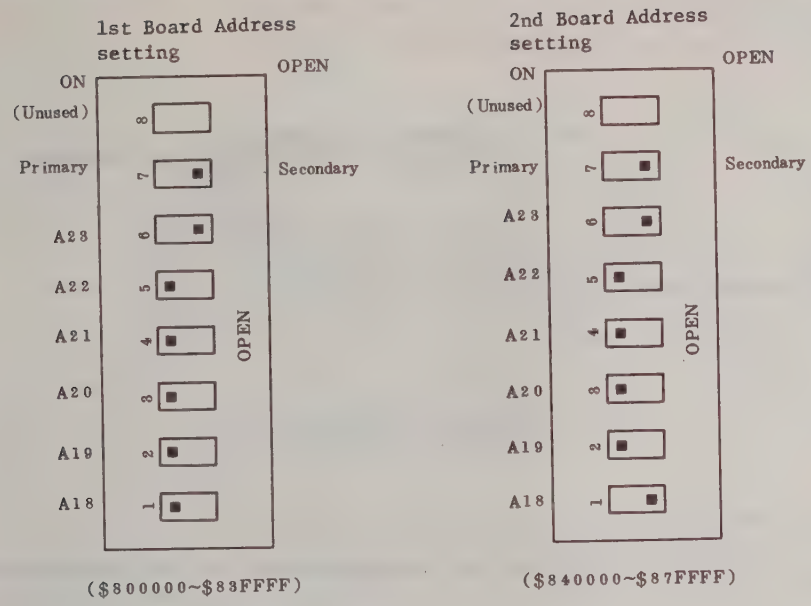


Fig. 3-7 Address Setting of Emulation Memory

Insert an emulation memory board in the emulation memory slot shown in Fig. 3-2. Either of the memory boards can be inserted in either of the slots. The procedure of installing the emulation memory is described in the following:

(1) See that the ASE power is off. (The POWER lamp on the right side of the ASE station front panel should stay off.)
Then turn off both the host system and console power switches.
(2) Disconnect the ASE station power cable.
(3) Disconnect all interface cables (HOST, CRT, and PRINTER) from the ASE station front panel.
(4) Remove the ASE station front panel. (See Fig. 3-1.)
(5) Check that the address setting on each memory board is correct.

(6) Insert the board in the first or third slot along the housing guides with the component-installed side up. Then set card pullers with a memory board inside two supports, and force the pullers in securely.
(7) Install the front panel.
(8) Connect the HOST, CRT, and PRINTER interface cables.

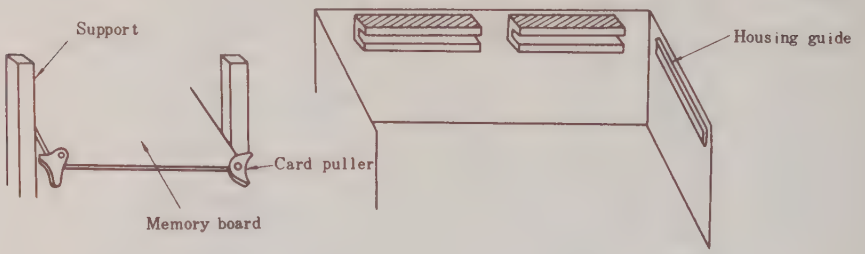


Fig. 3-8 Removing the Dust Cover

3.4 System GND Connections

A typical system GND connection that provides the best GND performance is shown below.

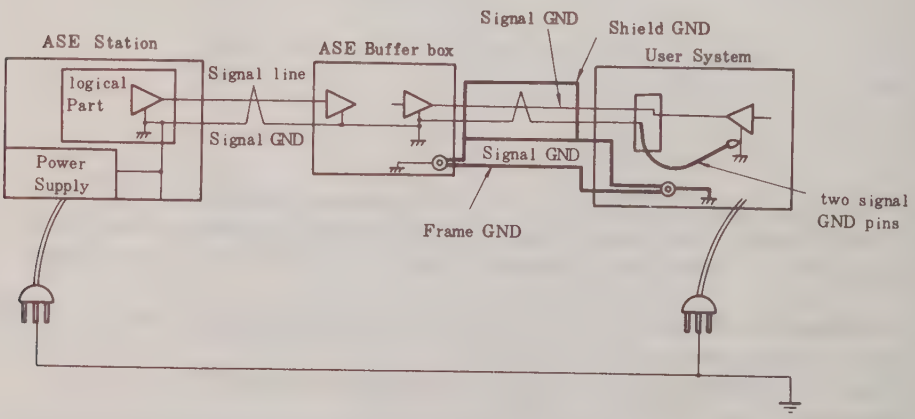


Fig. 3-9 Typical System GND Connection

[NOTE 1] Within the ASE station, the signal GND and the frame GND are connected to each other.
[NOTE 2] The buffer box signal GND is connected to the user system signal GND via the user system interface cable.
The GND clips from the user system interface cable give GND reinforcement.
[NOTE 3] For the user system, do not connect the signal GND to the frame GND but connect the frame GND alone. If it is difficult to separate the signal GND from the frame GND on the user system, ground it in a position as near the ASE frame GND as possible (if possible, the same position as the ASE frame GND).

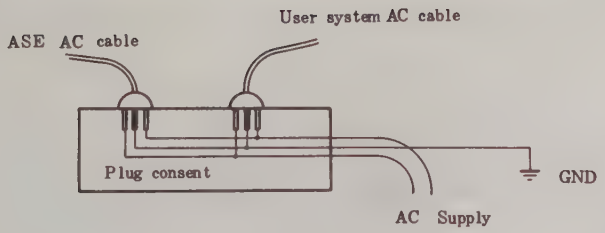


Fig. 3-10 Typical Frame GND Connection

It is important to provide appropriate connection for the user system under development to reduce noise, ground loop, and other adverse effects.

Using the GND cable (provided), connect the buffer box holder screws to the user system frame GND pin.

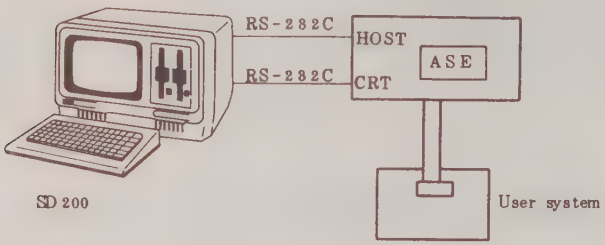
There are two GND clips extending from the user system interface cable. These must be connected to the IC GND pins near the MPU. Their purpose is to provide GND reinforcement.

The shield GND between the buffer box and the user system (see Fig. 3-9) can be obtained by using the jacket (provided). This GND protects the system from external noise such as static electricity. For its usage, see "3.2 User System Interface Cable".

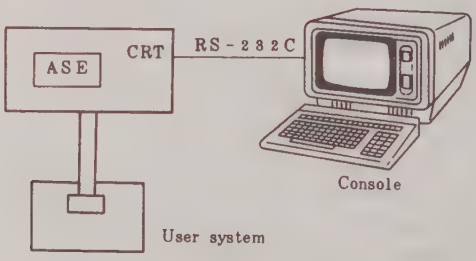
The ASE provides three debugging systems as follows:

(1) Interfacing with the SD200 as a host system
(2) Stand alone
(3) Interfacing with the general purpose host system, except the SD200.

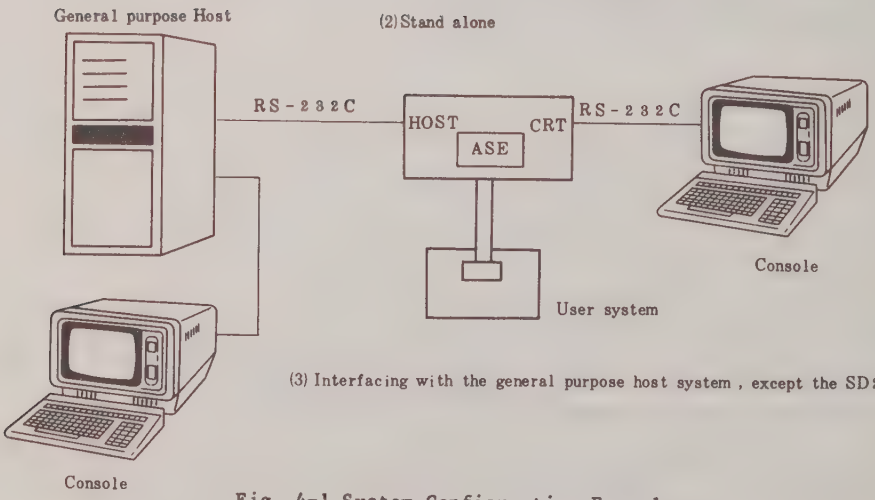
Fig. 4-1 shows three system configuration examples.



(1) Interfacing with SD 200



(2) Stand alone



(3) Interfacing with the general purpose host system , except the SD200

Fig. 4-1 System Configuration Examples

The ASE provides two serial interfaces (See Fig. 4-2):

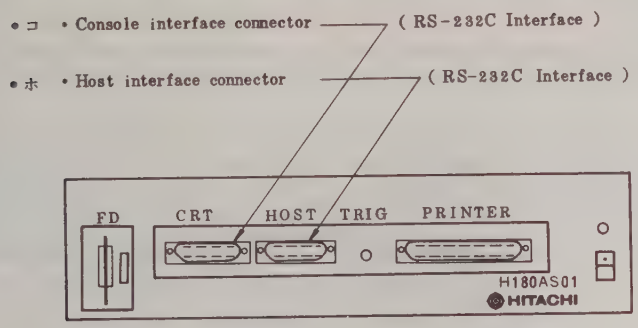


Fig. 4-2 Serial Interfaces on the ASE station

The console interface is used for key-input and screen display.

The host interface is used for transferring the object file.

For stand alone system, operates by connecting only the console through the console interface. When the object file is transferred using the general purpose host system, connect to the host interface.

When the SD200 is connected to the ASE, both the console interface and host interface are interfaced with the SD200.

4.1 Connecting with SD200

Connecting the ASE to the SD200 enables the object file which is prepared in the SD200, to be loaded into the ASE and to debug it.

Table 4-1 gives available commands when connecting the ASE to the SD200 host system. The ASE interface program must be activated so that these commands are available for the SD200 host system. (Refer to 9.21 HOST)

Table 4-1 Commands using the SD200

No.	Command	Function
1	LOAD	Loads a load module from the SD200 to the memory in the user system.
2	SAVE	Saves the contents of memory in the user system into the SD200.
3	VERIFY	Compares the load module on the SD200 with the contents of memory in the user system.
4	COMMAND_ CHAIN	Enters ASE commands one by one from files in the SD200 and executes them.
5	PRINT	Delivers the contents of a console display to the SD200 printer.
6	HOST	Activates the ASE interface program on the SD200.

4.1.1 Connecting procedure with SD200

The procedure for connecting the SD200 is shown as follows. (See Fig. 4-3.)

(1) Remove the short cable which connects the J3CR and J3DB on the rear panel of SD200.
(2) Take out the RS-232C Interface Cable marked "HOST" on the connector, connect one end of the cable to the connector marked "HOST" on the ASE station front panel, and the other to the "J3DB" on the SD200 rear panel.
(3) Take out the RS-232C Interface Cable marked "CRT" on the connector, connect one end of the cable to "CRT" on the ASE station front panel, and the other to the "J3CR" on the SD200 rear panel.

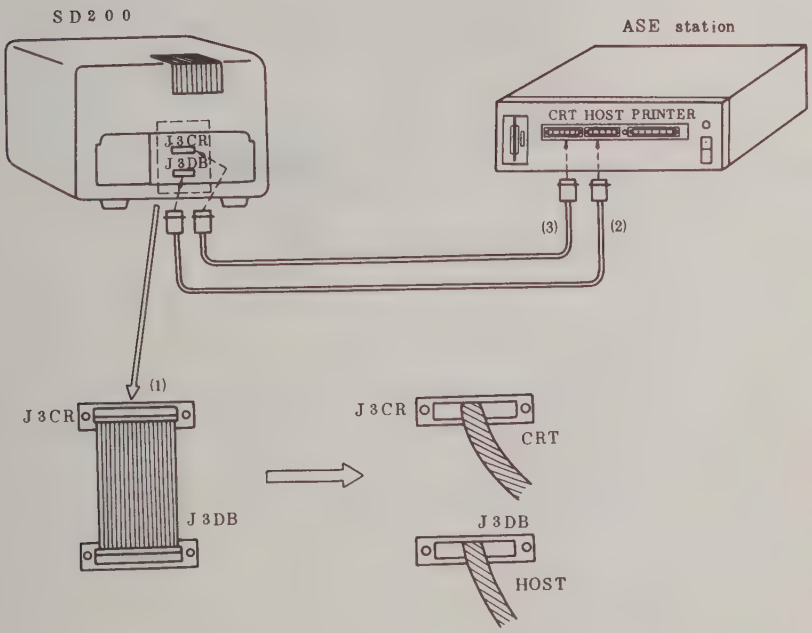


Fig. 4-3 Connecting with SD200

4.2 Connecting with Console

The ASE system operates independently by connecting with a console.

4.2.1 Console interface

Serial interface (RS-232C) is applied for input and output between the ASE station and the console. As shown in Fig. 4-4, the console connector marked "CRT" is provided on the ASE station front panel. Table 4-2 lists the connector pins and Fig. 4-5 shows their layout. When connecting, use the attached cable (which is marked CRT on the connector).

Fig. 4-4 Connecting with Console

Table 4-2 Console Connector Pins
(Connector name = CRT)

Pin No.	Signal Name	Pin No.	Signal Name
1	Frame Ground (FG)	14	Not used
2	Receive Data (RD)	15	Not used
3	Transmit Data (TD)	16	Not used
4	Clear to Send (CTS)	17	Not used
5	Request to Send (RTS)	18	Not used
6	Data Terminal Read (DTR)	19	Not used
7	Signal Ground (SG)	20	Data Set Ready (DSR)
8	Data Carrier Detect (DCD)	21	Not used
9	Not used	22	Not used
10	Not used	23	Not used
11	Not used	24	Not used
12	Not used	25	Not used
13	Not used		

Fig. 4-5 Console Connector Pin Layout

(a) Frame Ground (FG)

Connected to the ASE frame.

(b) Receive Data (RD)

Data receive line.

(c) Transmit Data (TD)

Data transmit line.

(d) Clear to Send (CTS)

With the CTS signal held at high level, the data is transmitted. During data transmitting, the CTS signal must hold at high level.

(e) Request to Send (RTS)

The RTS signal is held at high level with ASE power on.

(f) Data Terminal Ready (DTR)

The DTR signal is held at high level with ASE power on.

(g) Signal Ground (SG)

Signal GND

(h) Data Carrier Detect (DCD)

Holds at high level with ASE power on.

(i) Data Set Ready (DSR)

With this signal at high level, the system assumes that console power is on, enabling data transfer.

Table 4-3 shows the RS-232C interface level and Fig. 4-6 gives the console connection.

Table 4-3 Signal level

Level	Interfacing with RS-232C
High	+5 to +15V
Low	-5 to -15V

(Note) Figures indicate connector pin No.

Fig. 4-6 Connecting with console

4.2.2 Selection of RS-232C baud rate

Fig. 4-7 shows the location of the switches on the ASE control board (H180CONT01).

Fig. 4-7 Switches on ASE Control Board

SW 2 through SW 4 as shown in Fig. 4-7 are used to select a baud rate. The switch setup for each of the baud rates (600 to 9600BPS) is shown in Fig. 4-8.

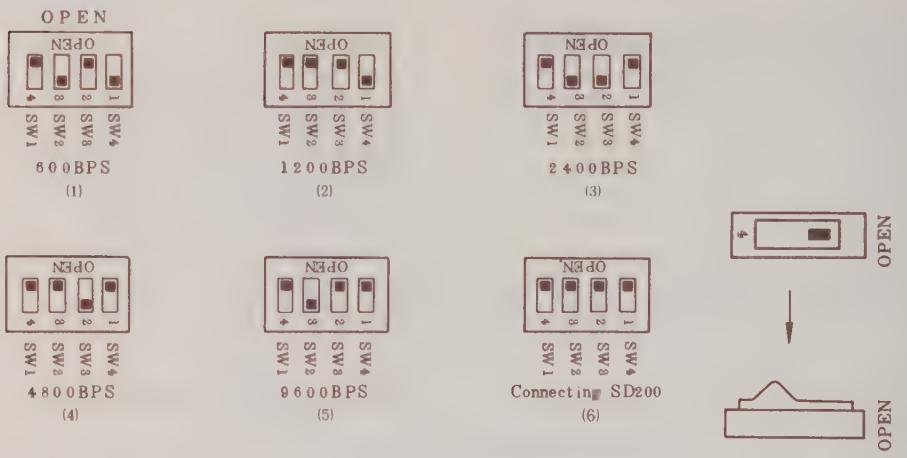


Fig. 4-8 Console Baud Rate Selection

(Note 1)

The ASE is set to (6) when it is shipped or delivered. Use setup (6) when connecting the ASE to the SD200.

(Note 2)

Use any setup (1) to (5) when connecting the ASE to the general purpose console.

(Note 3)

Set SW1 in the OPEN side in any connection. The RS-232C doesn't work if it is set in the ON side.

(Note 4)

Baud rate of the ASE should be the same as that of the console. If they are different, the system doesn't operate correctly.

Data format is shown as follows.

Start bit	1 bit
Data	8 bits
Stop bit	1 bit
Parity	None

4.2.3 Break key

The break key is used to interrupt a command execution in the ASE. This key specification is shown as follows. If there is no key matching this specification, the command cannot be interrupted.

O Break key specification

With the break key pressed, TD (Transmit Data) signal is held at low level during 100mS (min.) as shown in Fig. 4-9.

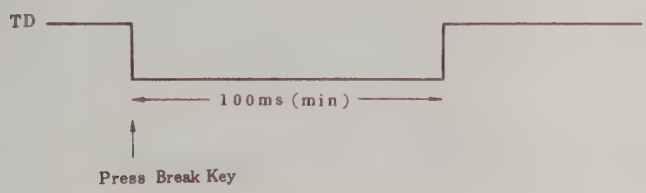


Fig 4-9 Break Key Specification

4.3 Connecting with General Purpose Host System

Connecting the ASE station to a general purpose host system allows the load module prepared by the general purpose host system to load and debug the user system.

The ASE provides commands which use the general purpose host system as shown in Table 4-4.

Table 4-4 Commands Which Use General Purpose Host System

No.	Command	Function
1	LOAD	Loads a load module from the general purpose host system to the memory in the user system.
2	SAVE	Saves the contents of memory in the user system to the general purpose host system.
3	VERIFY	Compares the load module on the general purpose host system with the contents of memory in the user system.
4	HOST	Connects the ASE with the host system.

4.3.1 Specification of host interface hardware

Specification of the RS-232C interface to connect the ASE with the general purpose host system is shown as follows.

(1) Signal level RS-232C
(2) Interface Full duplex
(3) Baud rate 300/600/1200/2400/4800/9600/19200 BPS
(4) Synchronization Start-Stop
(5) Start bit 1-bit
(6) Data bit 7/8-bit
(7) Stop bit 1/2-bit
(8) Parity E/O/None
(9) Protocol
- X - ON, X - OFF Control
- RTS, CTS Control

The specification is selected by the HOST command. For detailed selection, see " 9.21 HOST".

Input and output are done with the serial interface between the ASE and the host system. Fig. 4-10 gives the connection of the ASE with the host system. Interfacing is achieved through RS-232C. Table 4-5 lists the connector pins and Fig. 4-11 shows their layout.

Fig. 4-10 Connecting with Host System

Table 4-5 Host System Connector Pins
(Connector name = HOST)

Pin No.	Signal Name	Pin No.	Signal Name
1	Frame Ground (FG)	14	Not used
2	Transmit Data(TD)	15	Not used
3	Receive Data (RD)	16	Not used
4	Request to Send (RTS)	17	Not used
5	Clear to Send (CTS)	18	Not used
6	Data Set Ready (DSR)	19	Not used
7	Signal Ground (SG)	20	Data Terminal Ready (DTR)
8	Not used	21	Not used
9	Not used	22	Not used
10	Not used	23	Not used
11	Not used	24	Not used
12	Not used	25	Not used
13	Not used		

Fig. 4-11 Host System Connector Pin Layout

(a) Frame Ground (FG)
Connected to the ASE frame.
(b) Transmit Data (TD)
Data transmit line.
(c) Receive Data (RD)
Data receive line.
(d) Request to Send (RTS)
Used for control of data reception under RTS, CTS control. For details, see "4.3.2 Protocol".
(e) Clear to Send (CTS)
Used for control of data transmission under RTS, CTS control. For details, see "4.3.2 Protocol".
(f) Data Set Ready (DSR)
With the DSR signal held at high level, the ASE is considered to be connected to host system and data is transmitted.
Under X - ON, and X - OFF control this signal is not used.
(g) Signal GND (SG)
(h) Data Terminal Ready (DTR)
Stays at high level with ASE power on.

4.3.2 Protocol

The ASE provides RS-232C for serial interface and the following two software interfaces (protocol).

- X-ON, X-OFF control
- RTS, CTS control

These two control modes are described below.

(1) X-ON, X-OFF control
In operation of one word input/output with serial interface, the ASE handles data independently of the host system.
The following two points should be borne in mind with host computer in this mode.

(a) Buffering received data
If the host system can buffer data from the ASE, X-OFF ($13) is sent to the ASE to stop the data transfer from the ASE when the buffer becomes almost full. The host system, processing data until the buffer is empty, sends X-ON ($11) to the ASE to have data transfer resumed. If buffering is impossible, output X-OFF and X-ON signals at every data transfer.

(b) Transfer in LOAD and VERIFY command
The host system
① starts the record transfer and
② must check whether X-OFF signal has been sent when one record was transferred.
When transferring the next record,
③ receives X-ON signal.
And the host system repeats ① to ③. When it finishes transferring the last record, it closes the file.
In this mode, the serial interface is connected as Fig. 4-12.

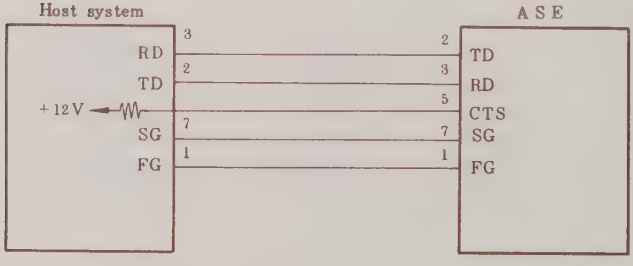
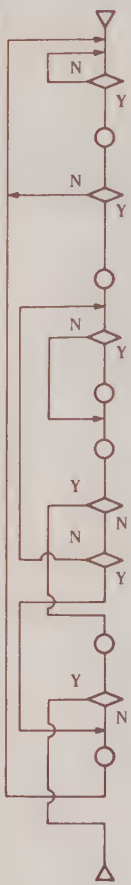


Fig. 4-12 Connection in X-ON, X-OFF Control

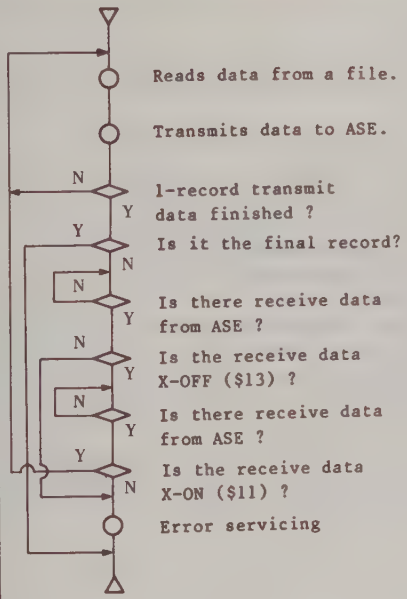
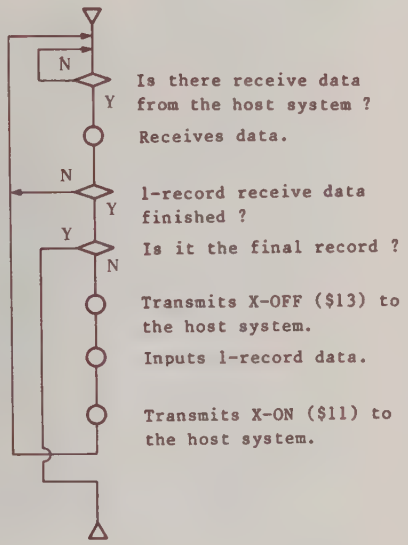
(c) Control code
In X-ON, X-OFF control, the following control codes are used.

Number	Control Code	Meaning	Direction
1	X-ON DC1($11)	Requests transmit of S-type and HEX-type records.	Host ←→ ASE system
2	X-OFF DC3($13)	Requests temporary stop of S-type and HEX-type transmit.	Host ←→ ASE systen

SAVE Command

Data Direction	Operation of the Host System	Operation of ASE Monitor
SAVE ASE ↓ Host system	 Is there receive data from ASE ? Adds the receive data to the buffer. Is the rest of the buffer less than 7 bytes ? Transmits X-ON ($13) to ASE ? Is there recieve data from ASE ? Adds the receive data to the buffer. Reads data from the buffer. 1-record data reading finished ? Is the buffer empty ? Outputs the record to a file. Is it the final record? Transmits X-ON ($11) to ASE.	 Is there receive data from the host system ? Is the receive data X-OFF ($13) ? Is there receive data from the host system ? Is the receive data X-ON ($11) ? Transmits data to the host system. All transmit data finished ?

LOAD, VERIFY Command

Data Direction	Operation of the Host System	Operation of ASE Monitor
LOAD VERIFY Host system ↓ ASE	 	

(Note) The ASE operates data up to CR ($OD) as one record.

(2) CTS, RTS control

"High" or "Low" of CTS and RTS described below represents the level of the cable connecting the ASE to the host system. In software the level depends on hardware of the host system, and usually the level in software is "High", that of hardware is "Low".

(a) One-record output

The ASE sends data and completes one-record transmitting process after checking that ASE CTS has become "Low".

(b) Transmitting in LOAD, VERIFY command

On receiving every record data, the ASE performs the following processes.

① The ASE sets RTS to "High" and prompts to the host system transmitting one-record data.

② After receiving one-record data from the host system, sets the RTS to "Low" and stops transmitting data temporarily.

Accordingly, in this system, it is necessary to pay attention to the following two points on the host system side.

(i) When receiving one-record data

- When the ASE can receive one-record data, sets RTS to "High", to make possible transmitting of the ASE.
- After receiving one-record data, sets the RTS to "Low" for 50 μS or more.

(ii) Transmitting in LOAD, VERIFY command

When transmitting one record, the ASE sends one-record after checking that the CTS signal has become "High". After transmitting, check that the CTS signal is "Low".

In this system, connect the serial interface as shown in Fig. 4-13.

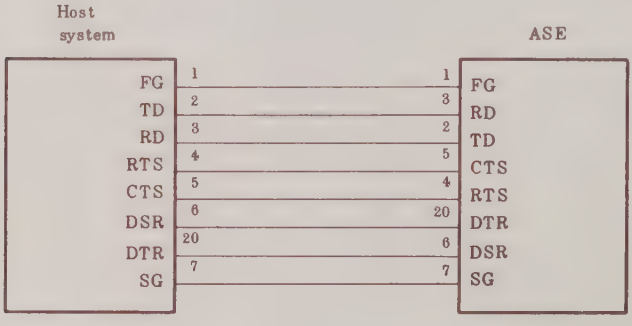


Fig. 4-13 Connection in CTS, RTS Control

When transmitting data, if the ASE can receive the data, the RTS signal is set to "High" level, and when not, RTS signal is set to "Low" level.

When receiving data, it is transmitted only when the CTS signal is at "High" level (refer to Fig. 4-14).

(c) RTS, CTS signal timing

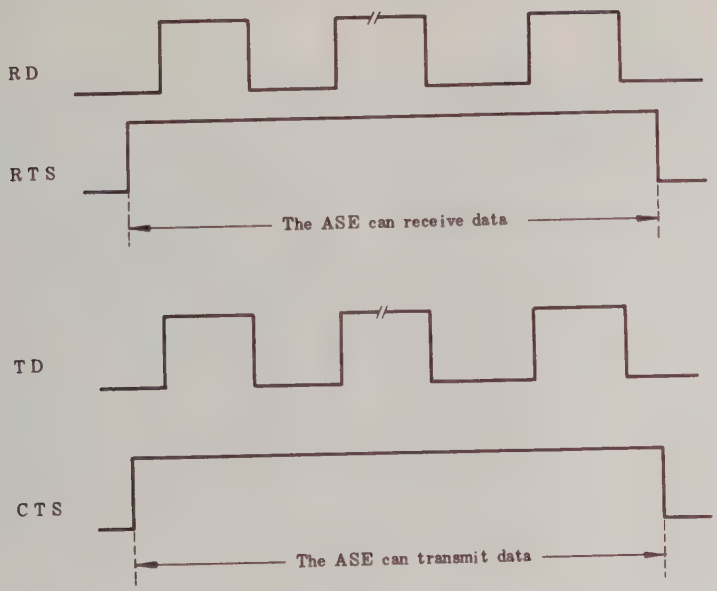
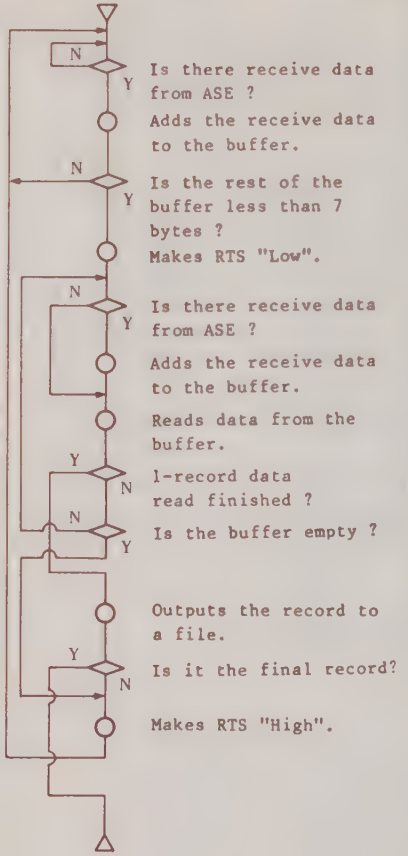
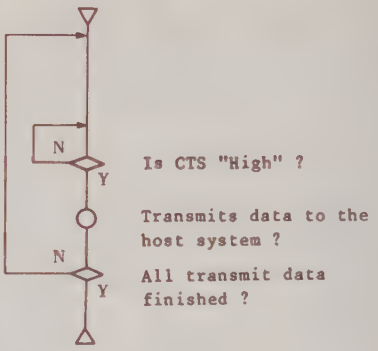
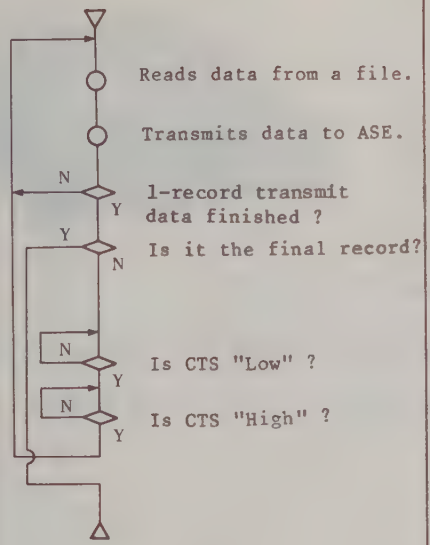
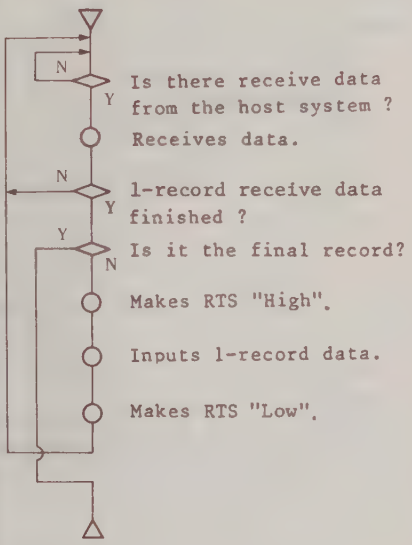


Fig. 4-14 RTS, CTS Signal Timing

SAVE Command

Data Direction	Operation of the Host System	Operation of ASE Monitor
SAVE ↓ ASE ↓ Host system		

LOAD, VERIFY Command

Data Direction	Operation of the Host System	Operation of ASE Monitor
LOAD VERIFY Host system ↓ ASE	 	

(Note) The ASE operates data up to CR ($0D) as one record.

4.4 Connecting with Printer

The ASE can be connected to the printer as shown in Fig. 4-15. Connecting between the ASE and the printer is based on the Centronics interface specifications. Use the printer interface cable provided for connection.

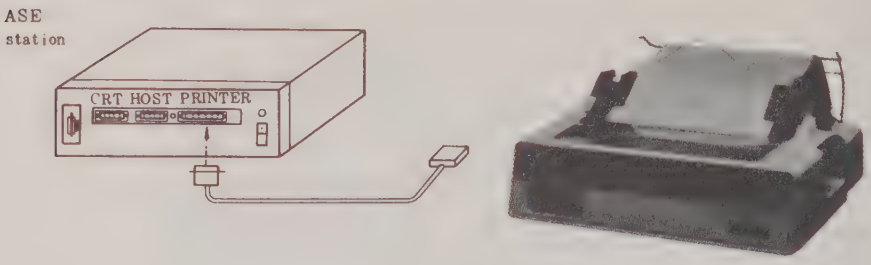


Fig. 4-15 Connecting a Printer

Connect a printer to the connector marked PRINTER on the ASE station front panel.
Table 4-6 shows the pin assignment between the ASE and the printer. Fig. 4-16 shows the printer connector pin layout.

Fig. 4-16 Printer Connector Pin Layout

Standard printer models that can be connected to the ASE are the M718 or the M5007 (parallel interface specifications) made by Brother Industries, Ltd. If using any other model, check its interface specifications.
The result of execution of ASE commands as displayed can also be printed on the host system side.
For the printer start command, see "9.28 PRINT".

Table 4-6 Printer Pin Assignment

Pair line No.	Signal name	Printer side pin	ASE side pin	Remarks
1	DATA STROBE-N	1	1	
	GND	19	20	
2	DATA1-P	2	2	
	GND	20	21	
3	DATA2-P	3	3	
	GND	21	22	
4	DATA3-P	4	4	
	GND	22	23	
5	DATA4-P	5	5	
	GND	23	24	
6	DATA5-P	6	6	
	GND	24	25	
7	DATA6-P	7	7	
	GND	25	26	
8	DATA7-P	8	8	
	GND	26	27	
9	DATA8-P	9	9	
	GND	27	28	
10	ACKNLG-N	10	10	
	GND	28	29	
11	BUSY-P	11	11	*29: soldered collectively on printer side.
	GND	29*	30	
12	PE-P	12	12	
	GND	29*	31	
13	SELECT-P	13	13	*30: soldered collectively on printer side.
	±0V	30*	32	
14	INPUT PRIME-N	31	14	
	GND	30*	33	
15	FAULT-N	32	15	
	±0V	16	34	
16	FG			• Cable shields and pair lines 16, 17 and 18 are collectively connected to FG.
	FG			
17	FG			
	FG			
18	FG			• Connected to pin 17 on printer side.
	FG			
Connector		57-30360 (manufactured by DDK)	D20419-21 DC24660 03-50641 DC-37P (all manufactured by JAE)	

The switch setting on the M5007 printer and the printer control code functions are listed in Table 4-7 and Table 4-8 respectively.

Table 4-7 Printer Switch Setting

Switch No.	Item	ON position	OFF position	Setting
8	AUT LF	Not AUTO LF	AUTO LF	ON
7	LF, VT, FF print	Not print command	Print command	OFF
5	CAN/DEL select	CAN valid	DEL valid	OFF
6	Home position	Return after print	No return after print	OFF
4	Power on select	POWER ON SELECT	POWER ON DESELECT	ON
3 2 1	Not used			ON or OFF

Table 4-8 Printer Control Code

No.	Item	Function
1	CR code function	Print command. Carriage stops after print.
2	FF code function	No print command. The paper is fed by the number of lines specified by the digital switch minus the number of lines already fed.
3	VT code function	No print command. The paper is fed by the number of lines specified by the code minus the number of lines already fed.
4	LF code function	No print command. The paper is fed by one line.
5	CAN or DEL code function	DEL code clears data in the buffer. CAN code generates a space.
6	Other special function codes	Details are found in the description below, "Printer operation control".
7	Power on select function	Provided.

M5007 printer operation control is described below.

(1) Printer operation control

The printer has a one-line buffer. It starts printing when it receives one of the print command codes (function codes), CR, LF, FF, and VT, after the entry of data to be printed. Otherwise, it prints data with buffer full (2176 columns).

(2) Special functions and their codes

(a) CR ($0D) and ($8D) ---------- Carriage Return

Print command. The carriage stops after print.
("Line Feed after Print" is also available if the appropriate printer specification is selected.)

(b) FF ($0C) and ($8C) ---------- Form Feed

Print command. After printing, the paper is fed by the number of lines specified by the code or digital switch minus the number of lines already fed. ("Form Feed after Print" is also available if the appropriate printer specification is selected.)

(c) VT ($0B) and ($8B) ---------- Vertical Tab
Print command. After printing, the paper is fed by the number of lines specified by the code minus the number of lines already fed. (The VT code can be changed to a non-print command if the appropriate printer specification is selected.)

(d) LF ($0A) and ($8A) ---------- Line Feed
Print command. After printing, the paper is fed by one line. (The LF code can be changed to a non-print command if the appropriate printer specification is selected.)

(e) ESC ($1B) and ($9B) ---------- Escape
Code extended. The printer takes the action determined by a combination of codes of 1 to 3 characters that follows the ESC code.

(f) BEL ($07) and ($87) ---------- Bell
The bell sounds for about 2 seconds.

(g) SO ($0E) and ($8E) ---------- Shift Out
Specifies expanded characters (5 characters/inch).
Up to 66 characters in the buffer can be expanded. (The 67th and subsequent characters will not be printed if entered.)

(h) CAN/DEL ($18) and ($98) / ($7F) and ($FF)
---------- Cancel/Delete (selected)
Clears the data in memory. No other signal causes clearing.

(i) DC1 ($11) and ($91) ---------- Device Control 1
Remote select.

(j) DC3 ($13) and ($93) ---------- Device Control 3
Remote deselect.

(k) AS ($17) and ($97) ---------- APL Selection
Specifies the APL character. Indicates that the character following the AS code is an APL code.

(3) Control codes expanded by the ESC code

(a) ESC + "B" + n (1 byte)
Valid when the digital switch indicates 00.
ESC + "B" specifies the page length. The one byte that follows it is used to set the amount of paper to be fed. Line feed is based on the value specified when the FF code was set; up to 255 lines can be set.
$1 \leqq n \leqq 255$ (binary code) One line is selected if n = 0.

(b) ESC + "C" + n(1 byte)
Sets 1/6 inches or 1/8 inches of line feed.
($06) ---------- 1/6 inches
($08) ---------- 1/8 inches

(c) ESC + "J" + n (1 byte)
Sets n/120 inches of line feed. $1 \leqq n \leqq 255$. If a single sheet of paper is used, the code sets n/108 inches of line feed.

(d) ESC + "H" + n (1 byte)
ESC + "H" specifies VT. The one byte that follows it is used to set the amount of paper to be fed. Setting is possible in up to 16 positions/page. The line feed unit is the same as for FF.

(e) ESC + "I"
Clears VT.

(f) ESC + "C" + ($00)
ESC + "J" + ($00)
Reset signal for the above (b) and (c). It is reset by either code (b) or (c). After resetting, the mode returns to digital switch selection.
Note: If the one byte that follows the ESC has not been defined as a control code, the two bytes including the ESC are assumed to be an undefined code.

(4) Rules for using digital switch codes when the FF is set

(a) Code setting is valid only when the digital switch is set at 00. (The digital switch is on the printer front panel.)

(b) If no code is set with the digital switch at 00, 66 lines will be automatically set when the power switch is turned on.

(c) If the code has been set with the digital switch at 00, but the switch indication changes to any value other than 00, then the code setting becomes invalid and the digital switch setting precedes. If the switch is returned to 00, the previous code setting becomes valid.

(d) The digital switch setting is read when;
(i) the power switch is turned on, or
(ii) FF starts.

(5) Rules for setting VT:
(a) Up to 16 positions can be set within the FF area.
(b) The VT code is not set when the power switch is turned on.
(c) With no VT set, the VT code causes the paper to be fed to the next FF position.
(d) The VT Set Clear causes the clearing of all positions.

(6) Priority of 6/8 SW, LINE 1/2 SW and LF code
(a) With code setting, the 6/8.LINE 1/2 SW is invalid.
(b) 1/6-inch LF is set when the power switch is turned on.

The timing chart of the support software in H18 0AS01 is shown below:

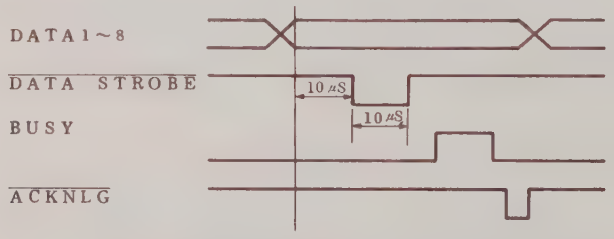


Fig. 4-17 Timing Chart of Support Software

NOTE: The support software checks the BUSY and $\overline{\text{ACKNLG}}$ signals. Be sure to connect the cable.

The timing chart of the M5007 printer is shown below:

(a) Data input (1)
Character data, space code, APL, and character data following input of SO:

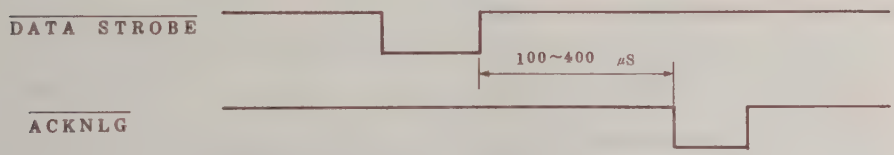


Fig. 4-18 Timing Chart of Data Input (1)

(b) Data input (2)
Undefined code, ESC sequence, SO, APL, and ESC sequence error:

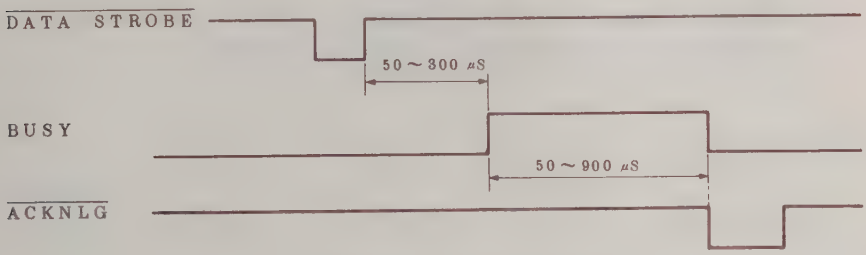


Fig. 4-19 Timing Chart of Data Input (2)

(c) Data input (3)
CAN, ALL space, CR, and CR alone:

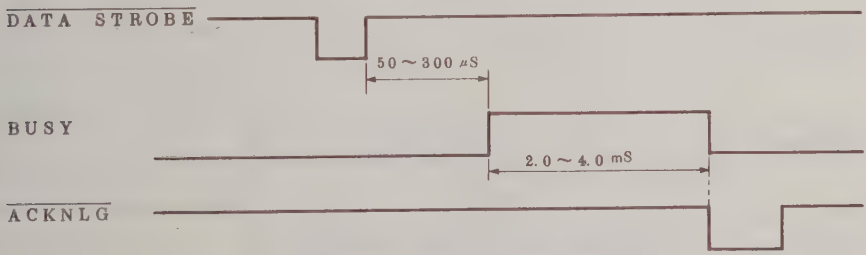


Fig. 4-20 Timing Chart of Data Input (3)

(d) Data input (4)
External operation occurs:

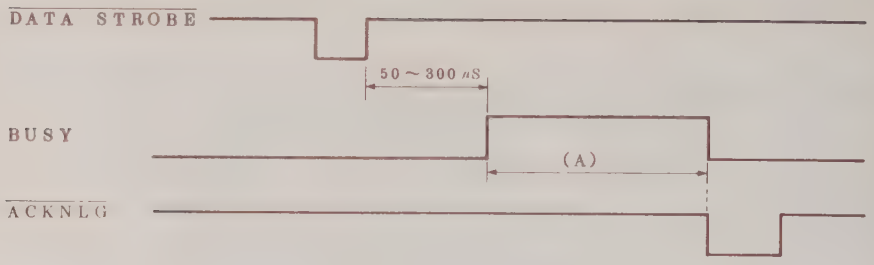


Fig. 4-21 Timing Chart of Data Input (4)

Value of (A) [reference]

Full line print	: 900 mS
BEL	: 2000 mS
LF (1/6 inches)	: 52 mS
VT (66 lines, 1/6 inches)	: 1360 mS (one form)
FF	: 1360 mS

(e) Data input (5)
DC3 operation:

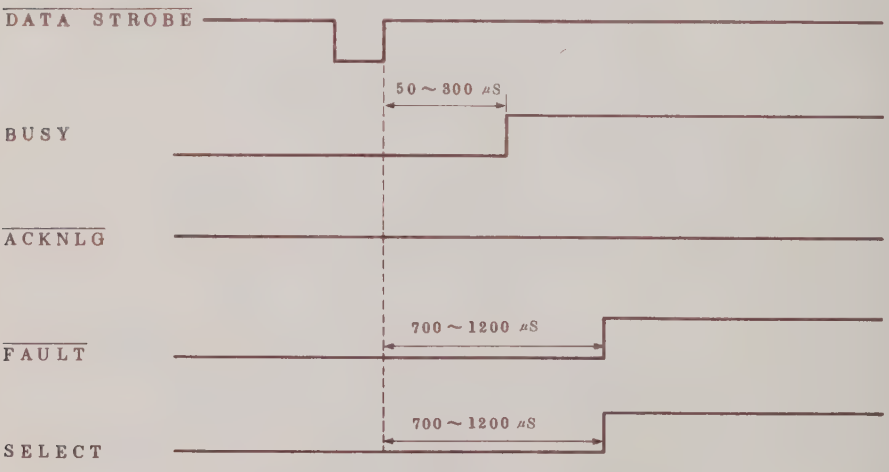


Fig. 4-22 Timing Chart of Data Input (5)

(f) Data input (6)
DC1 operation:

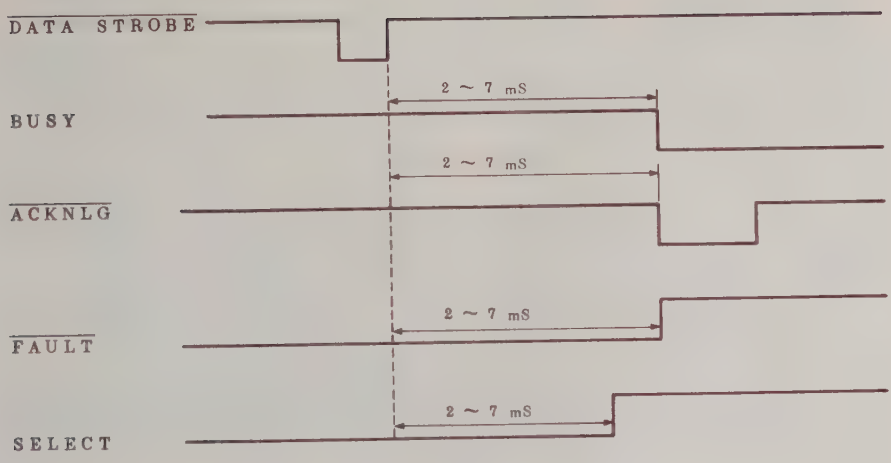


Fig. 4-23 Timing Chart of Data Input (6)

NOTE: Data input is measured at idling (SELECT ON).
If an encoder interrupt occurs, 100 μs maximum is added to the measured value.

5.1 Real-time Emulation

The ASE can perform real-time emulation (without a wait) to the MPU (HD64B180), at 6.144 MHz. The same thing can be said when using an emulation memory.

The user program starts with "GO" command. For easy debugging, the following modes can be selected.

. Trigger mode
(TRIG output pin outputs pulses as a synchronous signal for an oscilloscope)

. Cycle reset mode
(RESET is periodically input to the MPU when the user program is executed.)

5.1.1 Trigger mode

When using TRIG output pin on the front panel as a synchronous signal for the oscilloscope, pulses are output under hardware break condition (TRACE_BREAK_CONDITION command). In this case, as the user program doesn't stop, this mode is effective for debugging hardware (for more details, see the section of the trigger mode in "9.19 GO").

(1) Trigger signal output timing

When a hardware break condition is satisfied by TRACE_BREAK_CONDITION command, low level pulses are output from TRIG synchronously with ME or IOE of the bus cycle.

Fig. 5-1 shows the output timing on the TRIG output pin.

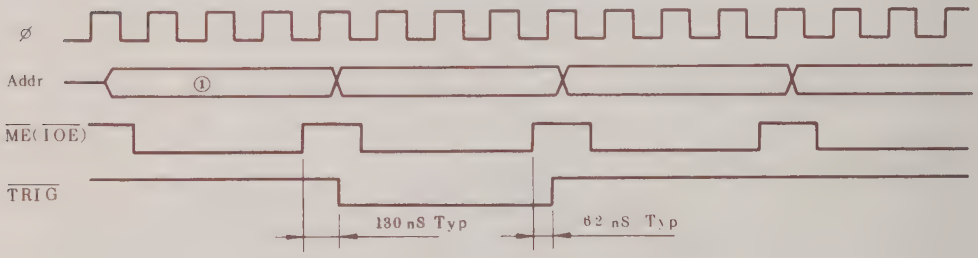


Fig. 5-1 Trigger Output Timing

Description : If the hardware break condition is satisfied during ①, a low level pulse is output after a typical 130nS from ME(IOE) negation. The trigger output is negated after a typical 62nS from the next ME(IOE) negation.

In the trigger mode, a low level pulse is output from the TRIG pin every time the hardware break condition is satisfied, and the user program doesn't stop.

If there are two hardware breaks, the low level pulse is output every time either is satisfied.

5.1.2 Cycle reset mode

Emulation starts by inputting RESET (this is called "cycle reset mode"). That is, RESET is periodically input from the ASE to the MPU when executing the user program. This is used to debug the system, such as observing waveforms in a specific period from the initialized state at power-on-reset (for more details, see the cycle reset mode in "9.19 GO").

When RESET is input to the MPU, low pulses are output at the same time from TRIG output pin, on the ASE front panel, just as in the trigger mode.

(1) Trigger signal output timing

In the cycle reset mode, the MPU is forcibly reset after a certain period specified by a command, ignoring the MPU condition. Fig. 5-2 shows the trigger output timing in the cycle reset mode. Press the BREAK key to stop the user program being executed in this mode.

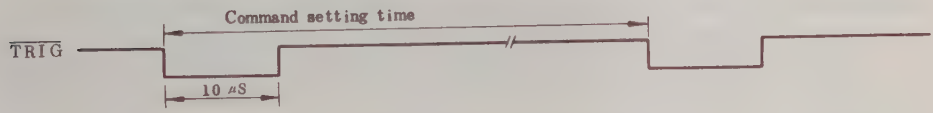


Fig. 5-2 Trigger Output Timing

5.2 Break

The user program can be stopped in the following ways;

(1) Hardware break

Any two hardware break conditions can be set in the following items by the ASE hardware break register (for more details, see "9.38 TRACE_BREAK_CONDITION" or "9.7 BREAK_MASK").

- Program counter
- Address bus
- Data bus
- External probes (8)
- Program counter, address area
- Setting the number of condition
- Program counter, address, data can be masked in the unit of one bit when using the BREAK_MASK command.

(2) Software break

When setting software break conditions, use BREAK command to change an instruction at a specific address in the RAM area to RST $30. The ASE stops executing the user program by executing RST $30. The specified address should be in the first op-code. Up to 255 points can be selected (See "9.6 BREAK").

(3) Key break

The ASE forcibly takes a break by pressing the BREAK key.

(4) Break timing using the external probe signal

Eight external probe signals can be set as a break condition. Whether the break condition is satisfied or not is detected by the HD64180 $\overline{ME}$ or $\overline{IOE}$ negation.

Fig. 5-3 shows the sampling timing by the external probe signal.

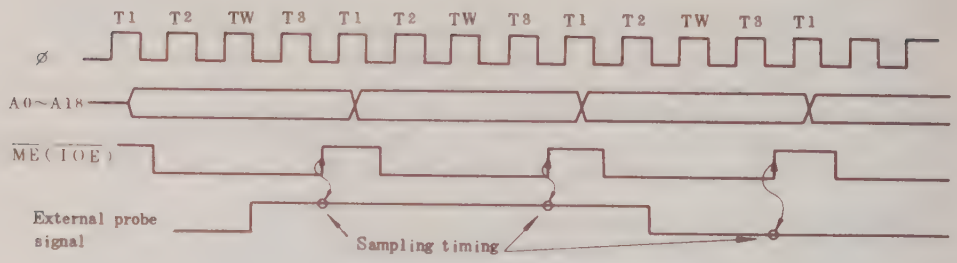


Fig. 5-3 Sampling Timing by External Probe Signal

Description : 1 Compare the external probe break condition with the external probe input signal at the ME or IOE rising edge.
2 If the comparison timing overlaps with a change in the external probe input signal, the break condition becomes indeterminate.

5.3 Trace

The ASE traces the MPU information combining GO and TRACE_BREAK_CONDITION commands during user program execution. Method, condition and display are described below.

(1) Trace information
2,048 cycles x 52 bits can be saved, whose information is as follows;

(a) Address bus	19
(b) Data bus	8
(c) External probe signal	8
(d) MPU input/output control signal	17

(2) Trace mode
There are two trace modes ; bus cycle trace mode and high-speed trace mode.

(a)In the bus cycle trace mode, the information in each bus cycle is saved in the trace memory. Fig. 5-7 shows the trace timing in this mode.
(b)In the high speed trace mode, the information is saved in the trace memory in each 125nS. Fig. 5-8 shows the trace timing in the high speed trace mode.

(3) Trace information display
The saved information is displayed as follows;
(a)The mnemonic of the instruction word in the trace buffer is displayed.
(b)The trace buffer contents are displayed in the unit of one bus cycle.
(c)The information traced in the high-speed trace mode is displayed.
For more details, see "9.37 TRACE".

(4) Trace condition

As a trace condition, either the trace start condition or the trace save condition can be selected (for more details, see "9.38 TRACE_BREAK_CONDITION").

(a) Free trace

Starts a trace when the user program execution starts with GO command and continues till a break takes place. If a trace condition is not set by the TRACE_BREAK_CONDITION command, free trace is automatically selected.

(b) Trace start condition

2,048 cycles of data after a trace condition is satisfied are traced. In this case, the user program stops when 2,048 cycles are traced.

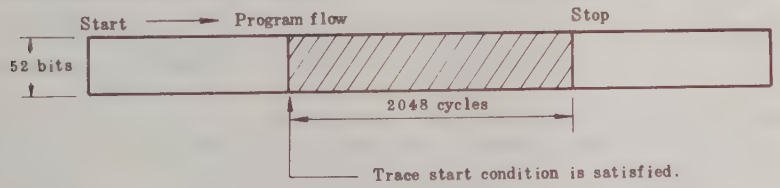
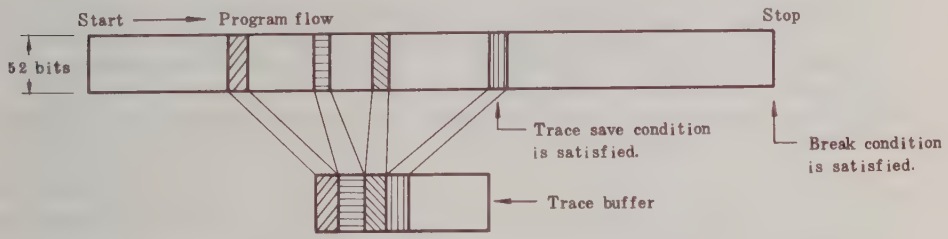


Fig. 5-4 Data Saving under Trace Start Condition

(c) Trace save condition

Data is traced only when a trace save condition is satisfied.



(Note) If the trace buffer is full, data can be traced under the trace save condition, where the oldest data in the trace buffer is deleted.

Fig. 5-5 Data Trace in Trace Save Condition

The trace save condition can be disabled, where the area not to be traced is traced.

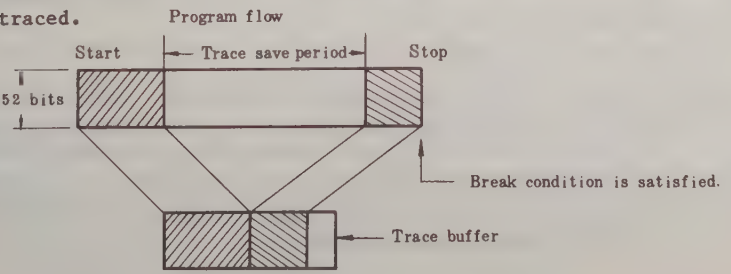


Fig. 5-6 Data Trace in Disable Condition

(5) Trace timing

(a) Bus cycle mode

In this mode, the address bus, data bus external probe signal, input/output control signal are saved into the trace memory in the unit of one bus cycle of the HD64180. The saved data is displayed with the TRACE command. The external probe signal is traced at the negate timing of the HD64180 $\overline{ME}$ or $\overline{IOE}$. Fig. 5-7 shows the trace timing in the bus cycle trace mode.

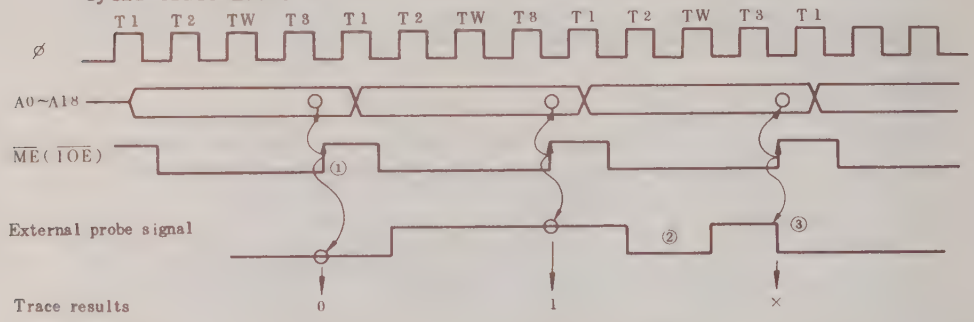


Fig. 5-7 Trace Timing in Bus Cycle Trace Mode

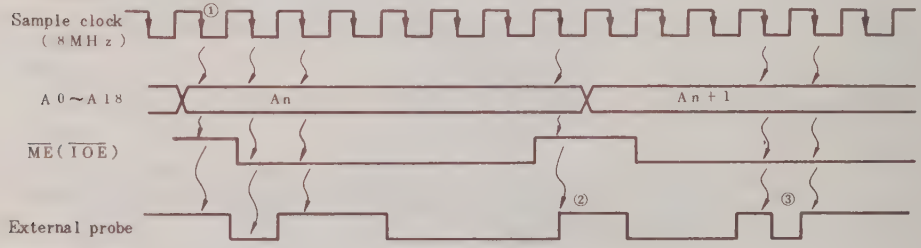
Des-cription : ① Sampled at the rising edge of $\overline{ME}$ or $\overline{IOE}$.(①)

② Any change of probe signals between samplig, which has the pulse width shown in Fig. 5-7, ②, will not be traced.

③ If the sampling edge overlaps with a change in probe signal, the trace data will be indeterminate.(③)

(b) High speed trace mode

In this mode, the address bus, data bus, external probe signals and input/output control signals are sampled and saved into the trace memory at 8MHz clock (every 125nS). The sample clock is asynchronous with the MPU clock. Fig. 5-8 shows the trace timing in the high-speed trace mode.



Trace results															
	A0~A18	An	An	An	An	An	An	An	An	An+1	An+1	An+1	An+1	An+1	An+1
	ME(IOE)	1	0	0	0	0	0	0	1	1	0	0	0	0	0
	External probe	1	0	1	1	0	0	0	×(②)	1	0	0	1(③)	1	1

Fig. 5-8 Trace Timing in High-Speed Trace Mode

Description: ① Sampled at the falling edge of the sample clock (8MHz).(①)

② If this falling edge overlaps with a change in the signal to be traced as shown in Fig. 5-8 ②, the trace data becomes indeterminate.

③ If input pulse of the probe input signal is less than 125nS, as shown in Fig. 5-8 ③, there may be a case where the data is not saved as trace information.

5.4 Single Step

(1) Displaying information in every single step

Emulation with a display of the following information every single step (For more details, see "9.34 STEP".)

- Mnemonic instruction
- Register contents

(2) Branch command display mode

Only the branch command is executed displaying the above information (For more details, see "9.34 STEP".)

(3) Stop condition

The following three are used, in which a single-step stop condition is set. (For more details, see "9.35 STOP".)

- Program counter
- Register
- Memory contents

5.5 Measurement of Execution Time

The GO command also displays the user program execution time from the start to the end caused by a break (for more details, see "9.19 GO" or "9.31 RESULT).

Maximum period = $2^{45}\mu$S (almost a year)

5.6 Memory Access

(1) Memory control

The ASE is normally fitted with an 8KB memory, and in addition, it can be fitted with 2 memory boards of 256KB optionally.
The HD64180 MPU has 512KB physical address. To control the memory, the physical address is divided into 128 segments (1 segment is 4KB): a user can decide the specifications (using or not using the user memory, emulation memory, write protect or not) for each segment. When specifying, see "9.24 MAP".

(2) Memory display, change or transfer

The ASE can display or change the user system memory or emulation memory. (For more details, see "9.25 MEMORY".)
The ASE can also transfer data between memories. (For more details, see "9.26 MOVE".) In particular, when changing a program such as an EPROM, it is very efficient to move the data in the user system memory to an emulation memory having the same address. (For more details, see "9.27 MOVE_TO_RAM".)

5.7 Monitoring Emulation

When executing emulation such as memory reference, memory change or user program execution, the ASE monitors the HD64180 in the buffer box using the control board in the ASE station.

Normally, emulation automatically ends when break conditions are met, however, the following are exceptions. In these cases, the emulation monitoring function shows the MPU state on the console. Emulation can be forcibly terminated by pressing the BREAK key. However, the ASE display nothing for the following (1) of undetected breakpoint.

(1) Breakpoint not detected

If a breakpoint is not detected, the user program is not terminated. Press the BREAK key to terminate the user program.

(2) $\overline{\text{WAIT}}$, $\overline{\text{BUSREQ}}$, $\overline{\text{RESET}}$ in "Low" more than 80μS.

If the $\overline{\text{WAIT}}$, $\overline{\text{BUSREQ}}$ or $\overline{\text{RESET}}$ signal is "low" more than 80μS, a time out is generated. When the ASE detects a time out every 300mS sampling, it displays what Table 5-1 shows on the console.

Table 5-1 Time Out Display

No.	Signal name	Display
1	WAIT	** TOUT S=x A=xxxx xxxxx
2	BUSREQ	** BUS REQUEST
3	RESET	** RESET

If $\overline{\text{WAIT}}$, $\overline{\text{BUSREQ}}$ or $\overline{\text{RESET}}$ goes "High" within 300mS, a time out is automatically reset. During a time out, pressing the BREAK key terminates the user program by master-resetting the HD64180 MPU (by input of RESET signal). For more details on the display, see "8.4 Full Screen Display Mode, ④ Display execution state".

(Note) Note that the display of No.1 in Table 5-1 will be also on the screen when parity error of emulation memory board occurs besides WAIT signal.

(3) SLEEP instruction during emulation

If the SLEEP instruction is executed and not canceled more than 80μS, the SLEEP flag is set. When the ASE detects the sleep flag in every 300mS sampling, it displays the following on the console;

** SLEEP

For more details on the display, see "8.4 Full Screen Display Mode, ④ Display execution state".

(4) User system power down

If the HD64180 clock from the user system is stopped by user system power down in the ASE command wait state or during user program execution, the ASE displays "USER SYSTEM NOT READY" and returns to the initial state.

5.8 Control and Condition Check of the User MPU

With the following functions, the user MPU can switch and inhibit the ASE internal signals. It can also check the normality of the signals and display their conditions, which is effective for hardware debugging.

(1) Selection of a clock

The ASE has a selecting function of a clock and the ASE internal clock (6.144MHz) of a user system. On starting the ASE when the user system is connected and the user clock is input, the user clock is selected, and when the user clock is not input, the ASE internal clock is selected.

In selecting the user clock, if the clock is stopped by power-down etc. from the user system, the ASE displays"USER SYSTEM NOT READY"and goes into the initial state.

In selecting the clock of the HD64180, each register of the HD64180 is initialized, as the ASE inputs and selects RESET to the HD64180. As for clock selecting by command, refer to "9.9 CLOCK".

(2) Control function of input signals

Enable/disable is specified to $\overline{\text{NMI}}$, $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, and $\overline{\text{INT2}}$, input signals of the MPU.

It is effective, when debugging without interruption from the user system. As for the specification method, refer to "9.22 IO_CONTROL".

(3) Checking the normality of I/O signals

On starting, the ASE checks the connection to the user system. Abnormalities such as a short-circuit in signal lines can be detected.

CHECK Command (As for the details of this command, refer to "9.8 CHECK".) faciliates the check of the user system interface line.

(4) Operating condition display of the ASE

The operating state of the ASE shown in Table 5-2 can be displayed. (As for details of the command, refer to "9.33 STATUS".)

Table 5-2 Displayed Operating State

Number	Contents
1	Clock type being used now.
2	Type of address input modes. (logical/physical address)
3	Existence of stop condition
4	The number of specified breakpoints
5	The number of registered symbols
6	Remaining size of the emulation memory
7	The condition of the RS-232C connected to the host system.
8	Existence of TRACE_BREAK_CONDITION command setting
9	Page number capturing the coverage information
10	Type of the base register for address calculation
11	Input of INT and NMI interruption (ENABLE/DISABLE)
12	Existence of trace capture of refresh cycles

(5) Display and update of the MPU register contents

MPU register contents can be displayed and updated. (For details of the command, refer to "9.41 .<register name>".)

5.9 Coverage

During user program execution, addresses accessed by the MPU are captured. As the capturable range is 64KB unit, the physical space (0 to $7FFFF) which can be accessed by the MPU is divided into 8 pages as shown below.

Page Number	Trace save range (physical address)
0	$ 0 to $ FFFF
1	$ 10000 to $ 1FFFF
2	$ 20000 to $ 2FFFF
3	$ 30000 to $ 3FFFF
4	$ 40000 to $ 4FFFF
5	$ 50000 to $ 5FFFF
6	$ 60000 to $ 6FFFF
7	$ 70000 to $ 7FFFF

For details, refer to "9.12 COVERAGE".

5.10 Software Development Support and Data Management

The function and the data management of software development and the support to improve the development efficiency are described as follows.

(1) Assemble

Specified assembly statement is converted into machine statement, and set into the memory of the user system. (For details of the command, refer to "9.3 ASSEMBLE".)

(2) Disassemble

Memory contents of the specified address are disassembled and instructions are shown as mnemonics. (For details of the command, refer to "9.13 DISASSEMBLE")

(3) Load, save, and verify functions of the user program (the host system)

The functions of the connection to the host system are as follows. (For details of the command, refer to "9.23 LOAD", "9.32 SAVE", and "9.40 VERIFY".)

- Loads the user program in the host system into the user system
- Saves the memory contents in the user system into the host system
- Verifies the user program in the host system and memory contents in the user system

(4) Load, save, and verify functions of the user program (3.5-inch floppy disk)

The following functions can be used between the inserted 3.5-inch floppy disk and the user system.

- Backup and verify the ASE system program
  (Refer to "9.4 BACKUP".)
- Format of 3.5-inch floppy disk
  (Refer to "9.15 FLOPPY_FORMAT".)
- Loads the user program from the 3.5-inch floppy disk to the user system
  (Refer to "9.16 FLOPPY_LOAD".)
- Saves the memory contents in the user system into the 3.5-inch floppy disk
  (Refer to "9.17 FLOPPY_SAVE".)
- Verifies the user program of the 3.5-inch floppy disk and memory contents of the user system.
  (Refer to "9.18 FLOPPY_VERIFY".)

(5) Screen divisional display function (Only when using the SD200)

The SD200 is connected as a console, then various data is displayed on the divided screen. (For details, refer to "8.4 Full Screen Display Mode", "9.14 DISPLAY".)

(6) Programmable function key (Only when using the SD200)

Commands are set corresponding to the function key. (For details, refer to "8.3 Special Key Function".)

(7) Command input using symbols

Debugging by symbol can be performed by loading a symbol file from the SD200 or registering a symbol from a console. (For details of commands, refer to "9.23 LOAD" and "9.36 SYMBOL").

(8) Numerical conversion

Specified data is converted into binary numbers, octal, decimal, hexadecimal, and ASCII characters. (For details of command, refer to "9.11 CONVERT").

(9) Address calculation

Logical address and physical address are calculated from the current MMU register value, and the specified value is displayed. (For details of the command, refer to "9.1 ADDRESS").

(10) HELP function (Display of all commands)

Command names and the meaning of all ASE commands are displayed on the screen. (For details of command, refer to "9.20 HELP").

5.11 User System Interface Circuit

The ASE is connected to the user system by the user system interface cable. Eight external probes are connected to the user system, and the tracing of probe signals and braking of probe signals can be accomplished.

The TRIG output pin is used in the trigger mode and cycle reset mode, and the output of a low level pulse is possible as a trigger signal for an oscilloscope.

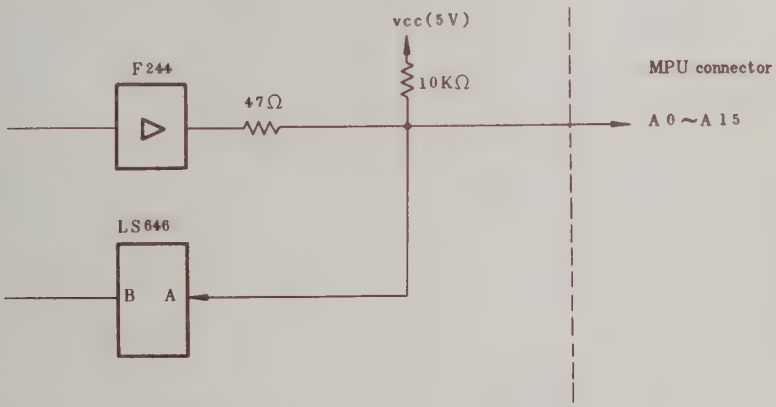
Each interface circuit is shown as follows.

(1) MPU connector interface circuit

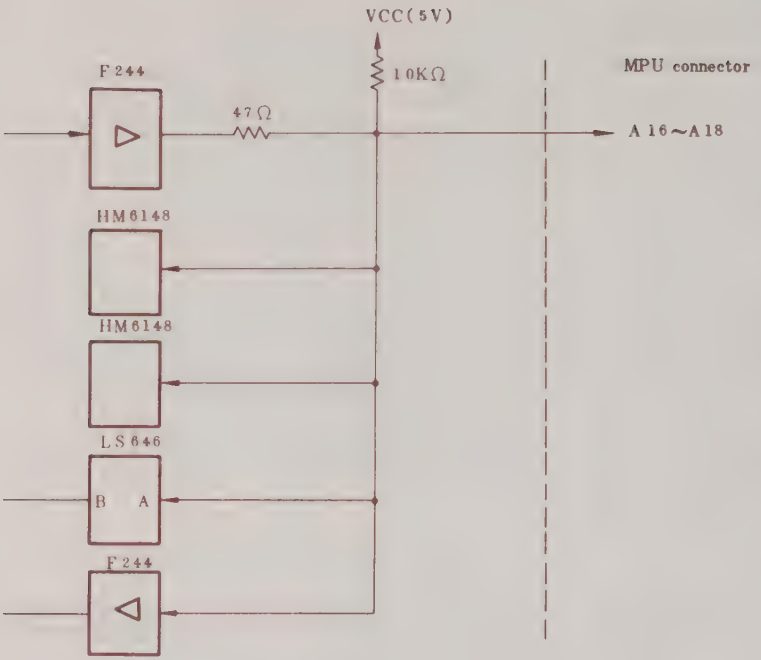
The interface circuit of the MPU connector of the ASE connected to the user system is shown as follows.

When connecting, adjust the hardware considering FANIN and FANOUT, the time delay of signal because of the buffer between the ASE MPU and MPU connector.

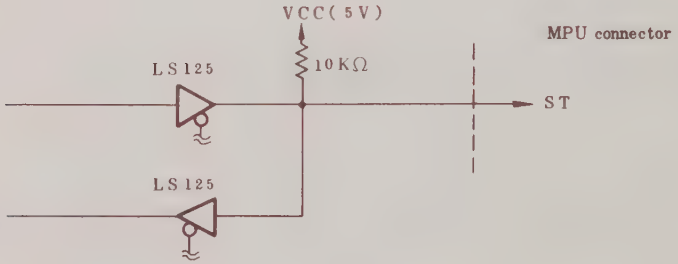
(A) Address bus (A0 to A15)



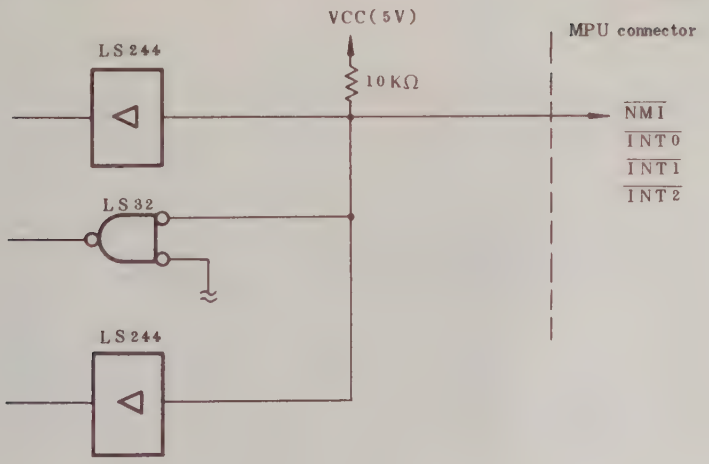
(B) Address bus (A16 to A18)



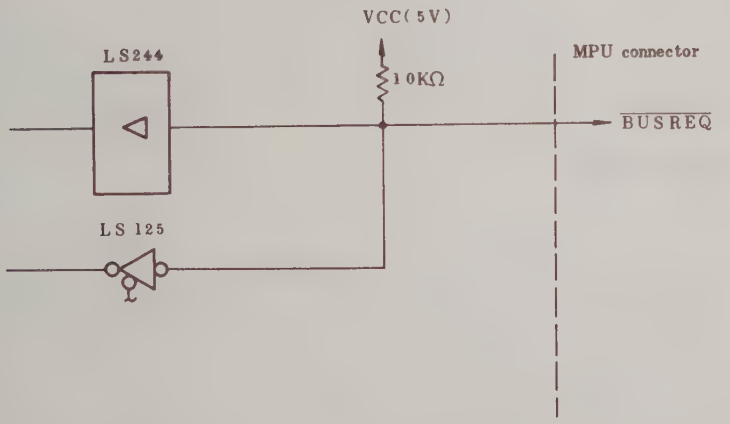
(C) System control signal (ST)



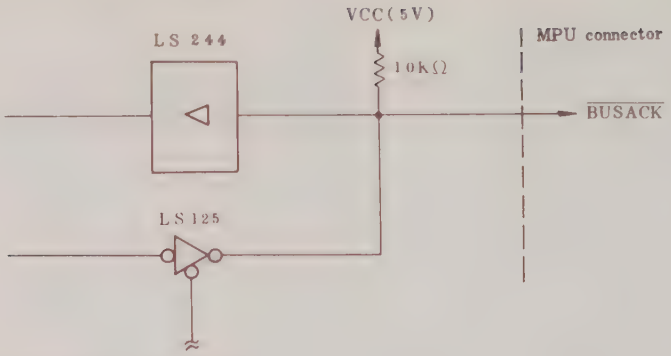
(D) Interrupt signal ($\overline{NMI}$, $\overline{INT0}$, $\overline{INT1}$, $\overline{INT2}$)



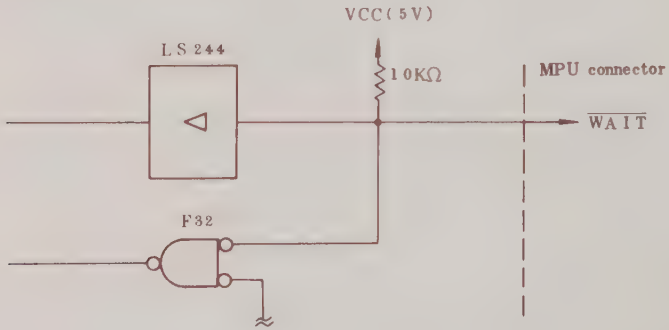
(E) System control signal ($\overline{BUSREQ}$)



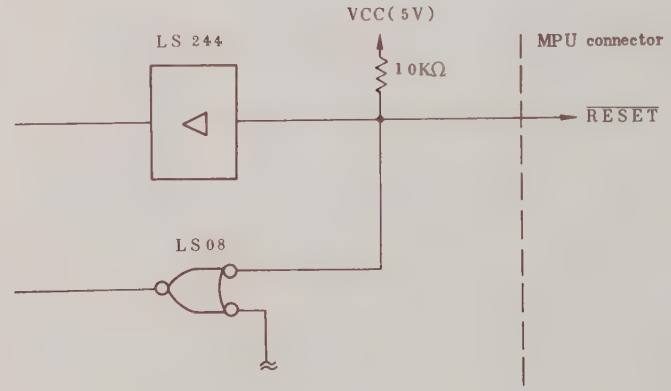
(F) System control signal ($\overline{\text{BUSACK}}$)



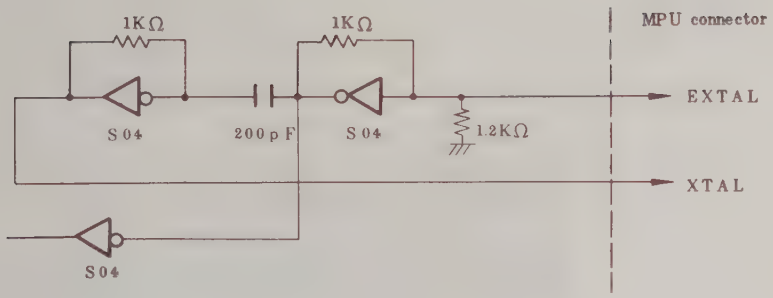
(G) Memory interface signal ($\overline{\text{WAIT}}$)



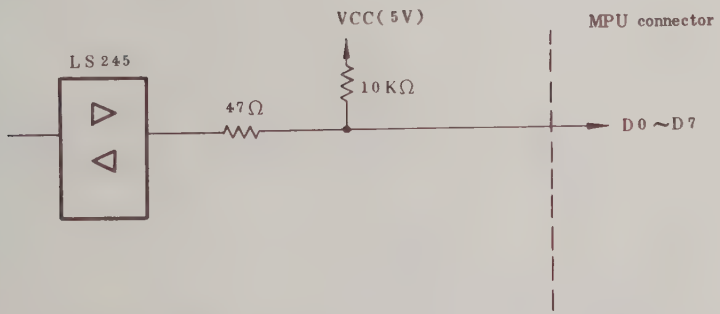
(H) RESET signal ($\overline{\text{RESET}}$)



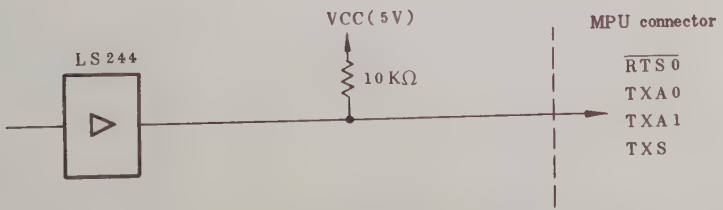
(I) Clock (EXTAL,XTAL)



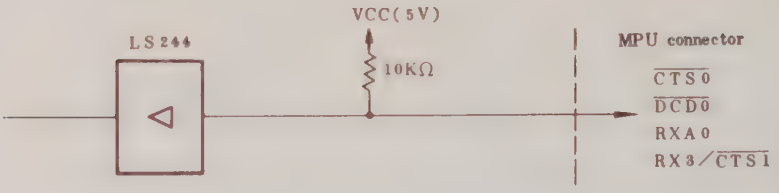
(J) Data bus (D0 - D7)



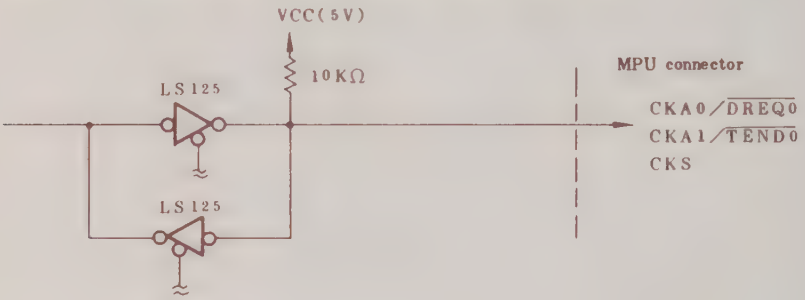
(K) Serial I/O signal ($\overline{RTS0}$, TXA0, TXA1, TXS)



(L) Serial I/O signal ($\overline{CTS0}$, $\overline{DCD0}$, RXA0, RXS/$\overline{CTS1}$)

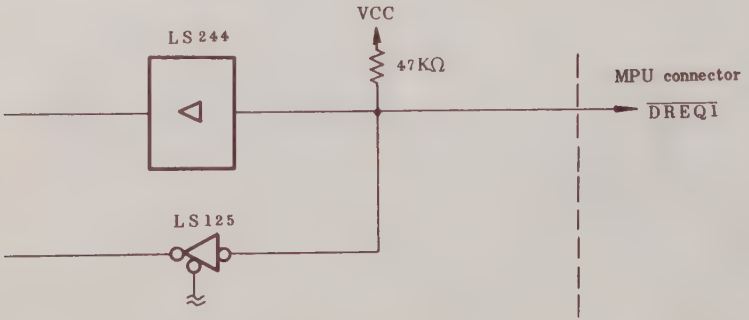


(M) Serial I/O, DMA signal ($CKA0/\overline{DREQ0}$, $CKA1/\overline{TEND0}$, CKS)

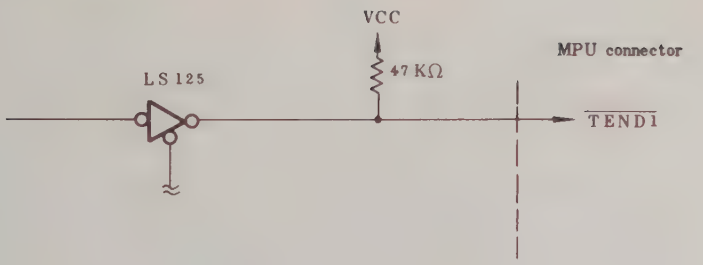


(N) Serial I/O signal (RXA1)

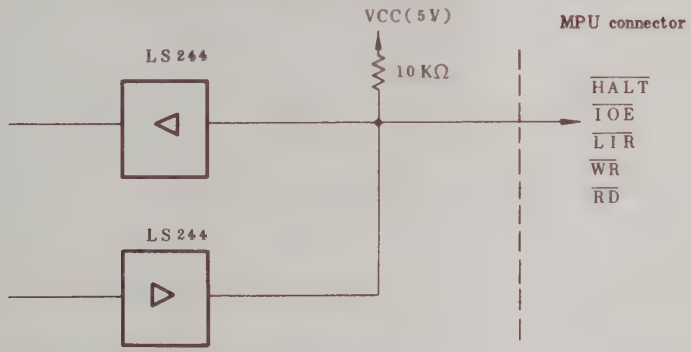
(O) DMA signal ($\overline{DREQ1}$)



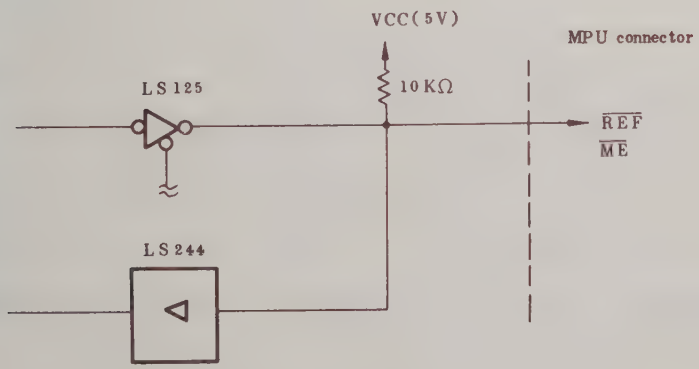
(P) DMA signal ($\overline{\text{TEND1}}$)



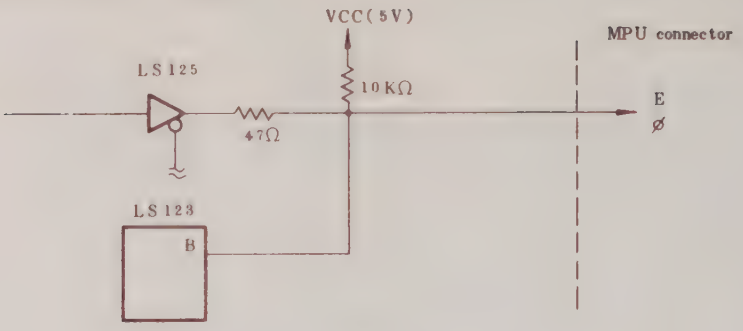
(Q) System control signal ($\overline{\text{HALT}}$, $\overline{\text{IOE}}$, $\overline{\text{LIR}}$, $\overline{\text{WR}}$,$\overline{\text{RD}}$)

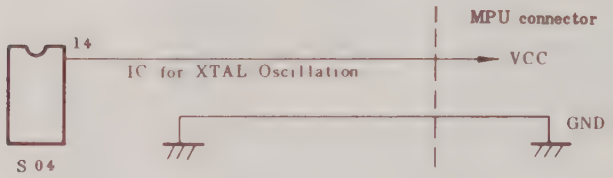


(R) System control signal ($\overline{\text{REF}}$,$\overline{\text{ME}}$)



(S) System control signal (E, ϕ)

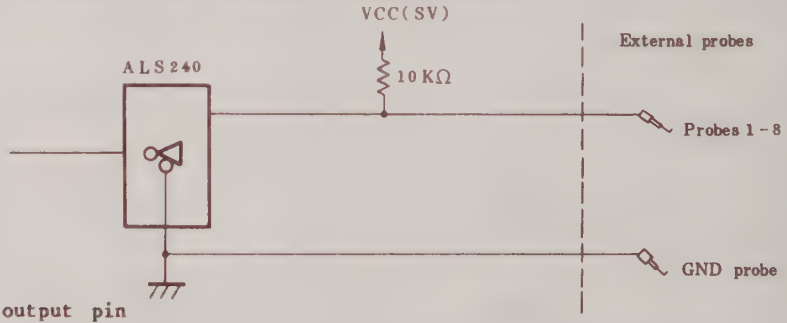




Power supply is provided from the user system.
Max.54mA

(2) External probe signal input circuit

As external probe signals, there are eight for the signal and one for GND, and they are shown in the following interface circuit.



(3) TRIG output pin

TRIG pin on the ASE station front panel is shown in the following interface circuit.

6.1 I/O Space Access

The HD64180 has 64k-byte I/O address space, in which there are internal I/O registers to support the HD64180 special functions such as MMU, DMAC, ASCI, CSI/O, refresh and timer functions.

The reference and change of I/O address space is done by MEMORY command. (Refer to section 9.25.)

(1) Reference and change of internal I/O register

The internal I/O registers are 64 bytes, but their location in I/O space is determined by the data in the I/O control register. In the ASE, base address of internal I/O registers is displayed at register display. (Refer to section 9.30.) It is obtained by adding the relative address of each I/O register to this displayed address. Refer and change by specifying the address with MEMORY command.

(Ex.) Change of MMU common base register (relative address $38)

```
: REGISTER
  ................  IR=CO --------I/O control regsiter
  ........................
: MEMORY F8;I
  00F8 000F8 00   ? 10       change of common base register
  00F9 000F9 01   ? 00
  00FA 000FA 00   ? .
```

(2) Reference and change of external I/O space

Addresses except for internal I/O registers are all external I/O address space. Refer and change with MEMORY command.

(Ex.) Reference and change of external I/O space

```
: MEMORY 160;I
  0160 00160 00 - ? 20
  0161 00161 02   ? .
```

6.2 Logical Address and Physical Address (MMU function)

As the HD64180 incorporates an MMU function, the address space is divided into logical address space and physical address space (for more details, refer to "HD64180 User's Manual") The ASE has specified commands (ADDRESS_MODE command) for each address so as to access by either of two address spaces. The address specified by each command operand is regarded as a kind of address specified by this command.

In the initial state, the ASE uses a physical address input mode.

(Ex.) Address mode of ASE command

Command input lines	Address
: MOVE 10500 11000 20000	Regards as physical address
: ADDRESS_MODE L	Becomes logical address input mode
: MEMORY 10 500	Regards as logical address
: MEMORY 20000	As it is over the range of logical address, it becomes an error.
: ADDRESS_MODE P	Becomes physical address input mode
: GO 10000	Regards as physical address

6.2.1 Address display at ASE command execution

Always, logical address and physical address are displayed in the address display at ASE command execution, such as MAP and MEMORY command.

Display format

xxxx xxxxx

① ②

① displays logical address in 4-digit hexadecimal.

② displays physical address in 5-digit hexadecimal.

(Ex.) :DISASSEMBLE 1000 1002

< ADDR >	<CODE>	<MNEM>	<OPRND>
1000 01000	210011	LD	HL, 1100

6.2.2 Logical address in physical address input mode

A logical address in physical address input mode is calculated from the value of MMU control register value when command is input.

Follow this calculating procedure.

(1) Calculate each physical address of common area 0, bank area and common area 1.

Physical starting address of the common area 0 Ⓐ = 0

Physical end address of the common area 0 Ⓑ = bank area lower limit address Ⓖ -1

Physical starting address of the bank area Ⓒ = bank area lower limit address Ⓖ + bank area base address Ⓘ

Physical end address of the bank area Ⓓ = common area 1 lower limit address Ⓗ + bank area base address Ⓘ -1

Physical starting address of the common area 1 Ⓔ = common area 1 lower limit address Ⓗ + common area 1 base address Ⓘ

Physical end address of the common area 1 Ⓕ = $FFFF + common area 1 lower limit address Ⓗ

MMU common/bank area register (I/O address $3A) contents

F	I
Ⓗ	Ⓖ

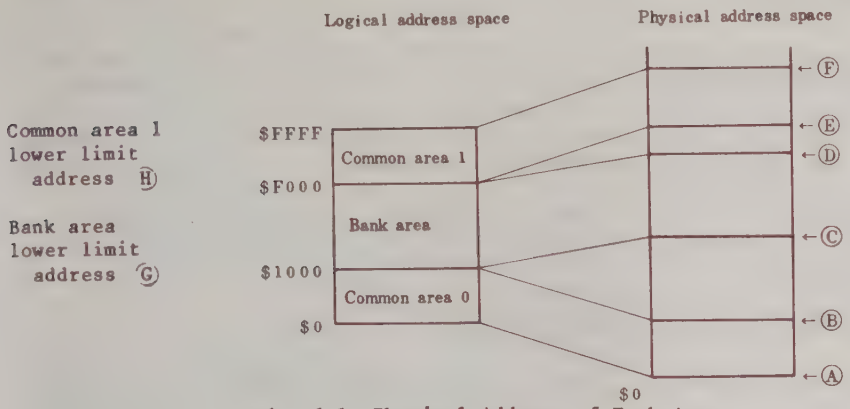


Fig. 6-1 Physical Address of Each Area

If physical address > 7FFFF, each area becomes following area.
Physical starting address<-->$7FFFF and
$0 -- (Physical end address - $80000),
or
(Physical starting address - $80000)<-->(Physical end address - $80000)

Fig. 6-2 shows an example when physical address > $7FFFF.

(a) Physical starting address < $7FFFF
Physical end address < $7FFFF

(b) Physical starting address < $7FFFF
Physical end address > $7FFFF

(c) Physical starting address > $7FFFF
Physical end address > $7FFFF

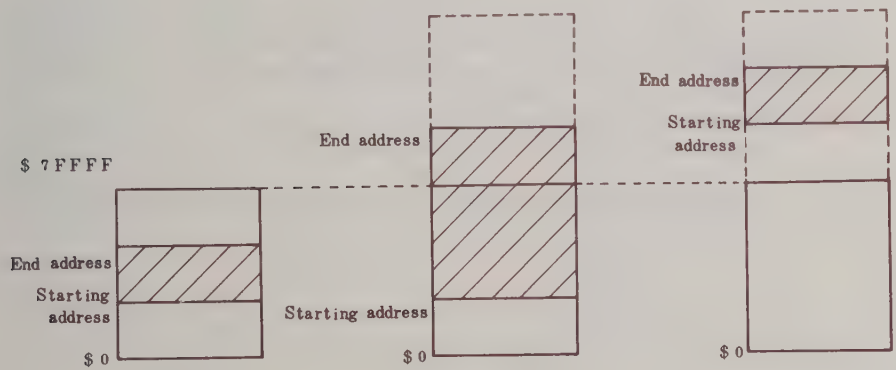


Fig. 6-2 Example When Physical Address > $7FFFF

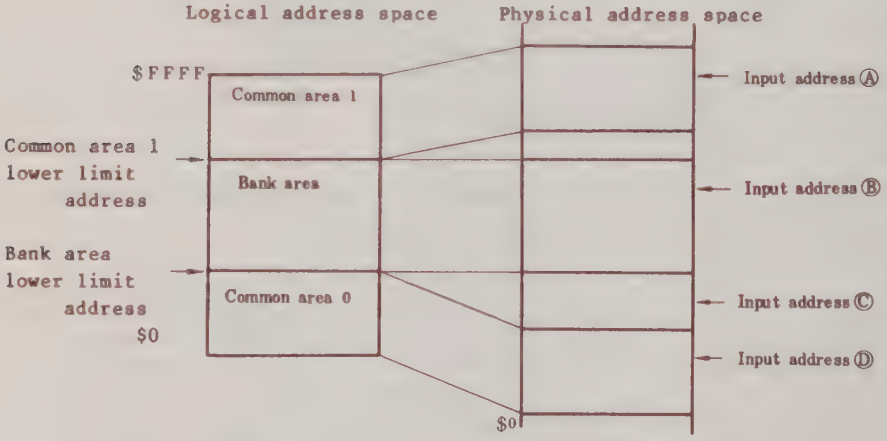
(2) Check that the input physical address is included in the area specified as the first priority base address by the BASE command. If it is included in the area;
The logical address = The input physical address - The first priority base address

(3) When the logical address isn't obtained by doing (2), calculates the logical address by the same method as (2) with the second priority base address instead of the first priority base address.

(4) When the logical address isn't obtained by doing (2) and (3), calculates the logical address by the same method as (2) with the third priority base address.

(5) When logical address isn't obtained by doing (2), (3) and (4), logical address is indefinite. When the logical address is indefinite, the ASE displays "****".

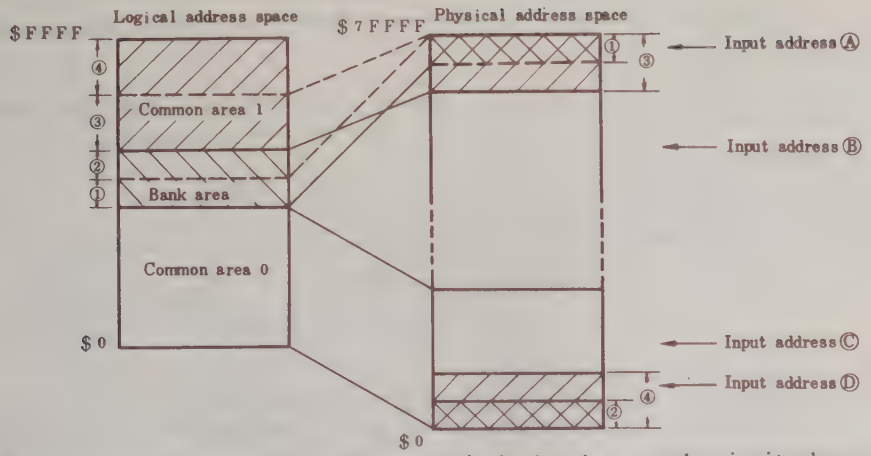
(Ex.1) When each area is not overlapped



Logical address of Ⓐ = Ⓐ - common area 1 base address
Logical address of Ⓑ = Ⓑ - bank area base address
Logical address of Ⓒ = indefinite
Logical address of Ⓓ = Ⓓ

Fig. 6-3 Logical Address Calculation When Each Area Is Not Overlapped

(Ex.2) When each area is overlapped



If the first priority base of BASE command is bank, the second priority base is common area 1;
Logical address of Ⓐ = Ⓐ - bank area base address (using the first priority base)
Logical address of Ⓑ = indefinite
Logical address of Ⓒ = Ⓒ
Logical address of Ⓓ = Ⓓ - common area 1 base address (using the second priority base)

Fig. 6-4 Logical Address Caluculation When Each Area Is Overlapped

6.2.3 Physical address in logical address input mode

A physical address in logical address input mode is calculated from the MMU control register value when command is input.

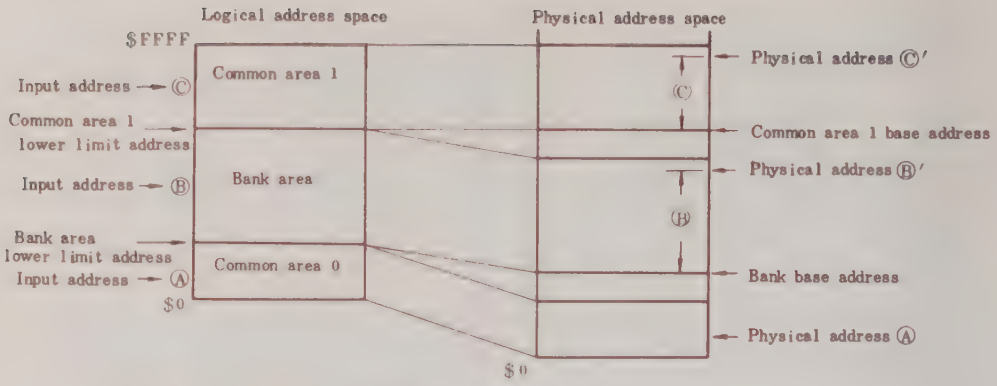
Follow this calculating procedure.

(1) If the input address < bank area lower limit address, the physical address = the input address.

(2) If the input address < common area 1 lower limit address, the physical address = the input address + the bank area base address.

(3) If the input address $\geq$ common area 1 lower limit address, the physical address = the input address + the common area 1 base address.

When a physical address is over $80000, the value subtracted $80000 from the physical address is considered as a physical address for calculation.



Physical address of Ⓐ = Ⓐ
Physical address of Ⓑ = Ⓑ + Bank area base address
Physical address of Ⓒ = Ⓒ + Common area 1 base address

Fig. 6-5 Physical Address Calculation in Logical Address Input Mode

6.2.4 Address range input by logical address

As mentioned in section 6.2.3, the address range specified by logical address isn't always a sequential address range of physical address, because base register changes the physical address. When the specified logical address range isn't a sequential physical address range, some commands become error. Fig. 6-6 shows the physical address range pattern corresponding to the logical address range, and table 6-1 shows commands to be noticed.

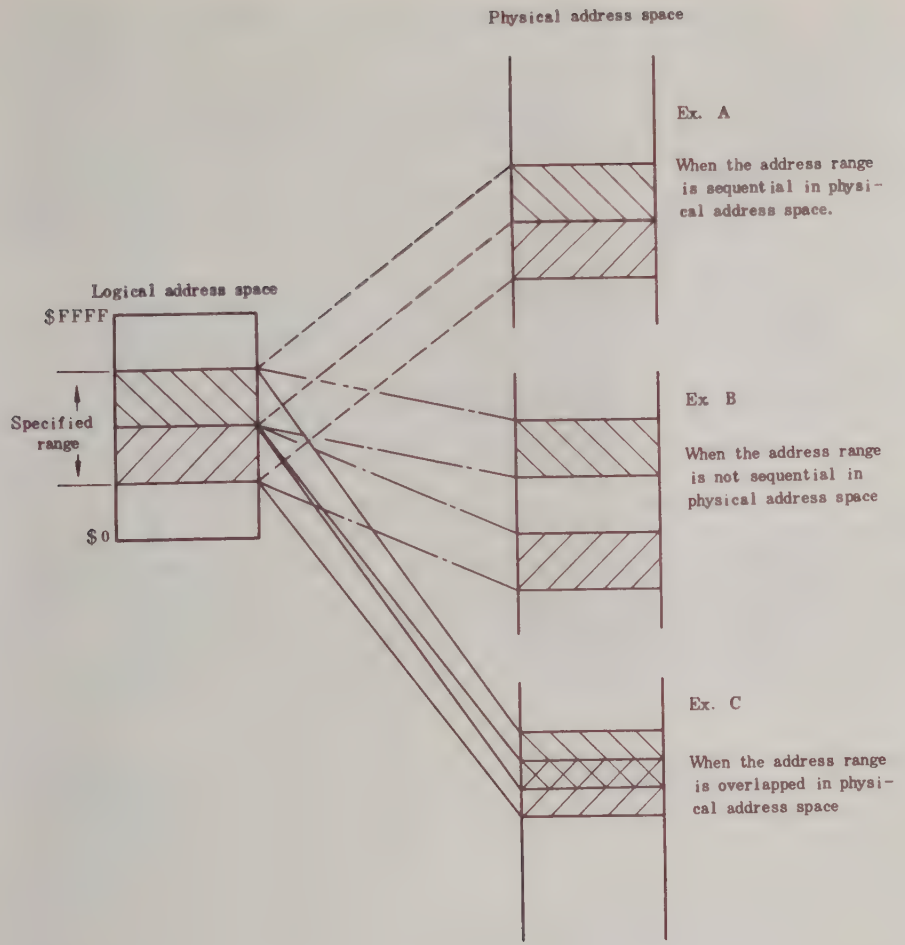

Physical address space
Ex. A
When the address range is sequential in physical address space.
Logical address space
$FFFF
Specified range
$0
Ex B
When the address range is not sequential in physical address space
Ex. C
When the address range is overlapped in physical address space

Fig. 6-6 Examples of Physical Address Range Corresponding to Logical Address Range

Table 6-1 Address Range in Each Command

No.	Command	Sequential	Not sequential	
		Ex. A	Ex. B	Ex. C
1	CHECK	o	x	x
2	COVERAGE	o	o	o
3	DISASSEMBLE	o	o	o
4	FLOPPY_SAVE	o	x	x
5	MAP	o	x	x
6	MEMORY	o	o	o
7	MOVE	o	x	x
8	MOVE_TO_RAM	o	x	x
9	SAVE	o	x	x
10	STOP	o	o	o
11	TRACE_BREAK_ CONDITION	o	o	o

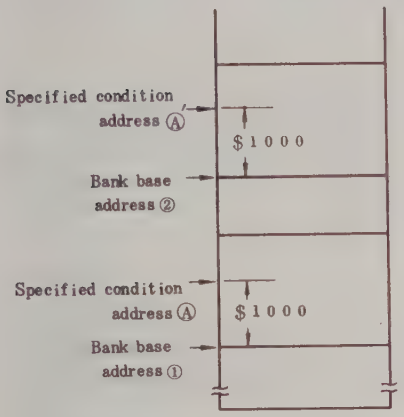
o : Operate normally.
x : An error occurs.

6.2.5 Logical address in TRACE_BREAK_CONDITION and STOP commands

In logical address input mode, the entered address is calculated to the physical address by the calculating method shown in section 6.2.3, and according to this value, the ASE performs each command operation, but, TRACE_BREAK_CONDITION (refer to 9.38) and STOP (refer to 9.35) commands are dealt with as follows:

TRACE_BREAK_CONDITION command specifies trace start condition, trace save condition and break condition. STOP command specifies stop condition of STEP command execution. An address is specified in these conditions. When the address input mode is logical address input mode, the ASE checks in a logical address input mode if the condition is satisfied.

```
(Ex.) Logical address in TRACE_BREAK_CONDIITON command
   : ADDRESS_MODE L          -- Set the logical adderss input mode
   : TRACE_BREAK_CONDITION
       :
    ? MA=1000                -- This condition is satisfied when $1000
                                in logical address is accessed.
```



In this case, the condition is checked in the logical address only.
So, as shown in the left, the physical address changes as the bank base address changes.(Ⓐ&Ⓐ')
address to satisfy the condition is changed.

Note) The bank area is from $0 to $1FFF (logical address).

Fig. 6-7 Specification of condition by logical address

6.3 DMA Function

The Direct Memory Access Controller(DMAC) is incorporated into the HD64180. The DMA operation is permitted only when user program is executed by GO command (Refer to section 9.19.). The DMA operation is interrupted in command input wait state or during command execution except for GO command.

(1) Change of DMA control register

The DMA control register can be changed by MEMORY command (refer to section 9.25.) as well as other internal I/O registers, but change of DMA status register (relative address $30) is inhibited. If required, change it by user program.

```
(Ex.) Change of DMAC control register
     :MEMORY 30;I
      0030 00030 00 ? 12
      *** ERROR 32 CANNOT CHANGE DMAC ST REG
      0030 00030 00 ?
      0031 00031 01 ? 02
      0032 00032 05 ?.
```

(2) Flow of DMA operation

Fig. 6-8 shows the relationship between operation of DMA and the ASE command.

Command	DMA operation
: GO START	DMA operation starts by user program
: RESULT	DMA operation is interrupted during ASE command operation
: STEP	
: GO	DMA operation restarts by GO command.

Fig. 6-8 Relationship between DMA operation and ASE command

(Note) The DMA operation always restarts regardless of the address where the operation starts by GO command. Also, when the DMA operation restarts, the ASE makes DME bit "Enable" just before user program execution. Therefore, when the user program restarts, some byte DMA operation has already executed.(Not included as a trace information.)

6.4 Refresh Function

The HD64180 outputs a refresh signal for dynamic RAM by operating a refresh control register. As a user memory contents is broken by interrupting this signal, it outputs continuously even when user emulation stops. The ASE outputs a refresh cycle to the user system with the power ON. The contents of the refresh control register at power ON is the same as at reset. After that, it outputs the refresh cycle as specified by the user.

6.5 Timer Function

The ASE doesn't have a special support for timer function. When starting the timer function by write to internal I/O register with MEMORY command or when starting it during the user program execution, either during the user program execution or command operation is in start-up and outputs TOUT signal to the user system.

However, timer interruption is inhibited except for user program execution by GO or STEP command. TOUT signal is multiplexed as A18 signal, but it is not saved as a trace information.

6.6 ASCI Function and CSI/O Function

As for the Asynchronous Serial Communication Interface(ASCI) function and the Clocked Serial I/O(CSI/O) function, the ASE doesn't have a special support. When starting the ASCI or CSI/O function by write to internal I/O register with MEMORY command or when starting it during the user program execution, either during the user program execution or command operation these functions are in start-up and output signal to the user system.

However, interruption is inhibited except for user program execution by GO or STEP commands.

Input/Output signal --- CKS, RXS, CTS, TXS, CKA1, RXA1, CKA0
RXA0, TXA0, $\overline{DCD0}$, $\overline{CTS0}$, $\overline{RTS0}$

6.7 Forced Input Reset Signal from ASE

The ASE might forcedly input a reset signal into the HD64180. When entering a reset signal, internal I/O registers of the user system are all initialized. After entering a reset signal, it is necessary to respecify an internal I/O register.

In the following cases, forcedly input a reset signal by the ASE.

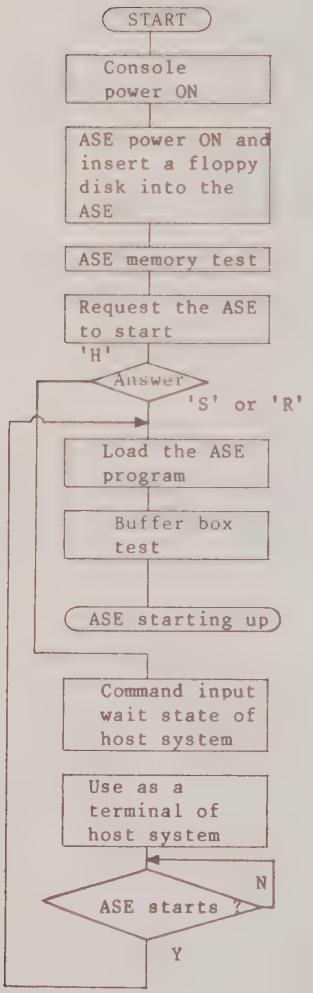
(1) When specifying cycle reset option (R) by GO command.
(2) When the CLOCK signal is changed by CLOCK command.
(3) When the execution stops by pressing the BREAK key during $\overline{WAIT}$, $\overline{BUSREQ}$ or $\overline{RESET}$ signal input as shown in section 5.7.
(4) When starting or terminating the ASE.

When the ASE tried to input RESET signal, the ASE outputs the following message except (1) above.

*** RESET IN BY ASE!

7. ASE STARTING UP

7.1 Starting Procedure



Display

① ASE MONITOR Vn.m
COPYRIGHTED BY HITACHI, LTD. 1985.
TESTING

② RAM 01234
SYSTEM MEMORY TEST

③ ASE SYSTEM MEMORY 012 -- (&xxxKB)

START ASE

④ S : START ASE
R : RELOAD & START ASE
H : HOST TERMINAL MODE
(S/R/H) ? ⑤ CR

⑥ SYSTEM LOAD

⑦ HD64180 ASE (S180ASE1F) Vn.m
COPYRIGHTED BY HITACHI,LTD. 1985.

⑧ BUFFER BOX TEST
TOTAL SYSTEM MEMORY SIZE &xxxKB
*** RESET IN BY ASE !

⑨ CLOCK = <Clock type to use>

⑩ :

⑪ ** HOST TERMINAL MODE ??
CR

⑫ ASE CR

① -- Starting message of the ASE monitor. "Vn.m" shows the version number of the ASE monitor.

② -- Tests RAM on the ASE. Tests the RAM dividing into 5 blocks. Each block displays a number from 0 to 4.

③ -- Tests an emulation memory in the ASE station. First, displays 0. Second, displays a number from 1 to 7 every time a test for 64KB is finished. Then displays a memory capacity of an emulation memory.

④ -- Displays the message asking if the ASE is started up or is used as a console terminal of the host system.

⑤ -- Inputs "S" or "R" when the ASE is started up or inputs "H" when the ASE is used as a console terminal of the host system. If input the other characters than "S", "R" and "H", the ASE goes to the state of ④ again.

⑥ -- In the case of "S" or "R" input, loads the ASE program from the ASE system disk already inserted. Note that if the ASE cannot load the program, it displays the following message and goes to the state of ④ again.

Table 7-1 Error Message When Loading the ASE Program

NO	Message	Description/solution
1	*** FD IS NOT SYSTEM FD ***	The floppy disk is not the ASE system disk. Insert the correct disk.
2	*** FD IS NOT READY ***	A floppy disk is not inserted. Insert a floppy disk.
3	*** FD I/O ERROR ***	The floppy disk is not read correctly. Insert another ASE system disk.

Note) The ASE program is loaded only when the ASE is started-up just after power ON. But, when "R" is specified, the ASE program is reloaded. Use "R" when loading test and maintenance program.

⑦ -- Starting message of the ASE program. "Vn.m" represents a version number.

⑧ -- Tests a memory in the buffer box.

The ASE goes to the state of ④ if there's something wrong with the ASE.

When the ASE is OK, it displays a total of emulation memory capacity in the ASE station and in the buffer box. You can use the total of emulation memory capacity.

⑨ -- Checks the connection to the user system. When a CLOCK signal in the user system is input, use the CLOCK signal in the user system, If the CLOCK signal is not input, use the 6.144MHz CLOCK signal of ASE. The ASE displays on the signal as follows.

CLOCK = <Using CLOCK type>

Using CLOCK type	USER -- CLOCK signal in the user system
	ASE(6MHz) -- 6.144MHz ASE CLOCK signal

⑩ -- Goes to the command input wait state. Afterward, you can input each command to the ASE.

⑪ -- In the case of "H" input as a response of requesting the ASE to start. Afterwards, you can use the console as a host system terminal until you input "ASE (CR) ". As for the SD200 system connection, however, once you press the RETURN key, the ASE displays "/" as a command input wait state of SD200 monitor or displays a floppy disk drive name and ">" as a command input wait state of CP/M-68K.

⑫ -- Please input "ASE (CR)" as a terminal mode when starting up the ASE. When the ASE program is not loaded yet, the ASE goes to the state of ⑥. When the ASE program is already loaded, the ASE goes to the state of ⑦.

Note) Be careful to input correctly a character one by one with the ASE start up command. Inputting "ASE (CR)" after revising by moving cursor does not start up the ASE. Please input again.

Note) After activating the ASE interface program (file name is ASE180.68K) on the SD200 with HOST command, the ASE interface program on the SD200 keeps operating when the ASE is forcibly finished with the power OFF. In this case, press the BREAK key power ON and select "terminal mode". Then press the character key "C" with the control key pressed. This operation terminates the ASE interface program on the SD200.

8. COMMAND INPUT AND DISPLAY

This chapter describes the input and display of ASE commands.

8.1 Command Syntax

Type ASE commands format as follows.

- Single-line command

  :<COMMNAD NAME> <OPERAND>;<OPTION> (CR)

- Interactive input format

  :<COMMAND NAME> ? (CR)

  :<RESPONSE REQUESTING MESSAGE> ? <RESPONSE> (CR)

  In interactive input format, a blank before "?" is omissible.

(1) Type in capital letters.

(2) Command name can be abbreviated. You can find full command names and abbreviations in the command description sections.

(Ex.) [L] O A D — Command name; LOAD; Abbreviation; L

(3) You can leave as much space as you like between words. "Words" means;
   (a) Command names
   (b) Punctuating symbols = > < * + – ; ( ) , $? space =< =>
   (c) Constants

(4) When entering word (;) (see (3)) following a space, the space can be omitted. (Only when each word can be clearly recognized.)

(Ex.) ASSEMBLE△; C (CR)
ASSEMBLE; C (CR)
Either of the above will be entered correctly.

(5) Enter <OPERAND> corresponding to commands.

(6) Enter constants as follows.

		ex.
(a) Binary	Start with "%".	%10111000
(b) Octal	Start with "@".	@726
(c) Decimal	Start with "&".	&396
(d) Hexadecimal	Start with "$", or nothing.	$FE56 EF56
(e) Character	Enclose with apostrophes.	'MOZI'

(Notes) • A constant must be within 4 bytes.
• Multinominals can be accepted as constants,
(Ex.) A+B, A-C.
• If you want to input an apostrophe as a constant, type two apostrophes and enclose them in another two apostrophes.
(Ex.) '''' --- One apostrophe constant ($27)
Note that constants will be all regarded as a positive number.

(7) Registered symbol names can be entered as constants. A symbol name is within eight alphanumeric symbols whose start is always a letter. First type quotation marks to distinguish it from other strings.
(Ex.)

```
:MOVE   "ABE 1000  2000
        ⇓
:MOVE   100  1000  2000
```

The same meaning as the command below because symbol name ABE is defined as $100.

Note that the value of a symbol name is a physical address. Therefore, when referring to a symbol name in logical address input mode, the ASE calculates a logical address from a physical address of a symbol name, then take the logical address as a constant.
At this time, if the logical address is infinite, it becomes error.

(8) If you specify any character of the row starting with "*" as a character of a command, the ASE takes it as a comment and goes to command input wait state.

(9) Up to 78 characters can be entered, but they include CR and LF codes. So the user can enter up to 76 characters in fact.

8.2 How to Input One-line ASSEMBLE Command

(1) Format

Enter an ASSEMBLE command as follows.

<OP-CODE>△<OPERAND1>,<OPERAND2>ⓒR

<OP-CODE> ---- Mnemonics described in "HD64180 User's Manual"
<OPERAND> ---- Operands include the following.

reg:	Direct register reference
(reg):	Indirect register reference
(IX+constant) (IY+constant)	Indirect reference of the index register with displacement
Constant:	Constant reference
(Constant):	Address contents reference
$±constant:	Relative address reference
($±constant):	Reference of relative address contents

(Notes) · You can enter a symbol name as a constant.
· For constants, refer to the constant entering method in Section 8.1 on the previous page.

(a) You cannot add any label or comment to the command.

(b) Constant format differs from the SD200 CP/M-68K 64180 cross macro assembler.

```
(Ex.) LD  IY, $AFE0    (ASE command)
      LD  IY, 0AFE0H   (SD200 CP/M-68K 64180 cross macro assembler)
```

(c) Put "$" immediately before hexadecimal numbers. 'A', 'B', 'C', 'D', 'E', 'BC', 'DE' without "$" are understood as registers.

```
(Ex.) INC (IX+A)   (WRONG)
      INC (IX+$A)  (RIGHT)
```

(d) Display a machine code with the instruction in address $0 if option C (modification of memory contents) is not selected or if the logical address is indefinite in physical address input mode.

```
(Ex.) LD  IY,($+2) ---- Displays the same machine code set by the
                        below command, since "$" means $0 without
                        C option specified.
      LD  IY,(2)
```

(1) Backspace key (Control Key + 'H')

Pressing a control key and the character "H" at the same time is regarded as a backspace key. When the backspace key is input, the cursor moves backward one character and the character is deleted. If you use the SD200, "←" key has the same function.

(2) Single-line cancel key (Control key + 'X')

Pressing a control key and the character "X" at the same time is regarded as a single-line cancel key. When the single-line cancel key is input, the cursor returns to the starting place in the line and the characters already input in the line are deleted. If you use the SD200, DEL LINE key has the same function.

(3) BREAK key

Pressing a BREAK key stops the command execution and the ASE goes to the command input wait state. In this case, the ASE displays "BREAK!". However, during real-time emulation with GO command, the ASE displays "BREAK KEY" as a reason of stopping GO command.

(4) Function key (F1 to F12)

Only the SD200 has the function keys. The function keys F1 to F12 have each command registered. Therefore, a command can be input with a function key without typing a command name. Each key has the following command registered in starting the ASE.

Table 8-1 Commands in Function Keys

No.	Key	Registered Command
1	F1	BREAK
2	F2	BREAK_MASK
3	F3	COVERAGE
4	F4	GO
5	F5	MEMORY
6	F6	QUIT
7	F7	REGISTER
8	F8	STEP
9	F9	STOP
10	F10	SYMBOL
11	F11	TRACE
12	F12	TRACE_BREAK_CONDITION

(Ex.)

Display	Procedure
:_	① In command input wait state, press the function key (F5).
↓	
:MEMORY_	② The registered command is displayed.
↓	
:MEMORY 100_	③ Revise the displayed command. When there's no problem, you don't have to do anything.
↓	
:MEMORY 100 ⓒR	④ Pressing the RETURN key causes to input the command.

Fig. 8-1 Example of Function Key Operation

(5) Registering in a function key (Shift key + function key)

You can modify the contents registered in a function key. Registering procedure is as follows. Note that only the SD200 has the function keys.

(Registering procedure)

① Press a shift key and a function key at the same time.
② The console displays "Fn-", then input a command will be executed.
③ Pressing a RETURN key checks whether the input command is of the ASE or not. If it is not of the ASE, an error occurs.
④ If the input command is of the ASE, the input line of the command will be registered.
⑤ When specifies S or P option with DISPLAY command (refer to section 9.14) at the SD200 system connection, the last line of the registered contents display is modified in registering the function key. Note that the command is displayed in the abbreviation format.

Note that the ASE checks only the command name in registering. The ASE does not check the operand, so be careful to input.

(Ex.)

Display	Procedure
:	① In command input wait state, press the shift key and the function key (F12) at the same time.
↓	
:F12-_	② The number of the function key to register is displayed.
↓	
:F12-M 100;W_	③ Input the command line to register.
↓	
:F12-M 100;W (CR)	④ The line is registered in pressing RETURN key.

Fig. 8-2 Function Key Registeration

8.4 Full Screen Display Mode

When the SD200 console is connected, the screen displays by specifying either P option (full screen display page mode) or S option (full screen display scroll mode) with DISPLAY command.

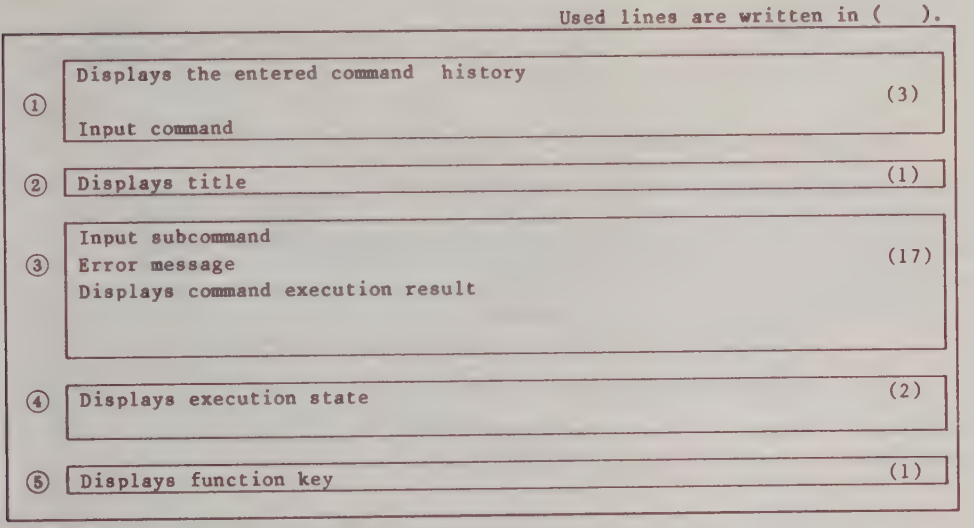


Fig. 8-3 Display Format in Full Screen Display Mode

① Input command and display the entered command history (3 lines)
3 lines are scrolled up including a command in the third line.
The upper 2 lines display entered commands history.

② Displays title (1 line)
A title displayed in command execution result display area is displayed.
This area is cleared when the next command is entered.

③ Subcommand input, error message display and command execution result display (17 lines).
It is used as a subcommand input, error message display, command execution result display area. P option (full screen display page mode) stops when 17 lines are displayed and, the next 17 lines are displayed when ⓒR is entered. In S option (full screen display scroll mode), lines are scrolled up without stopping the display.

④ Display execution state (2 lines)
first line

```
user system state  CLOCK=x MODE=x SP=x BR=&xxx SYM=&xxxx M=&xxxKB
      (a)           (b)     (c) (d)      (e)     (f)      (g)
```

second line

```
HOST=xxxxx TR-BR-COND=xxxxxx COVERAGE=x BASE=xx INT=x NMI=x REF=x
   (h)       (i)           (j)          (k)     (l)   (m)     (n)
```

(a) user system state
Displays user program execution state. The contents are as follows:

(i) **RUN PC=xxxx xxxxx ... Displays that the user program is being executed. Displays the value of the program counter.

(ii) **RESET ...RESET signal is entered into user MPU.

(iii) **SLEEP ....Displays SLEEP state.

(iv) **BUS REQUEST ....BUS REQUEST signal is entered into user MPU.

(v) **TOUT S=x A=xxxx xxxxx ..Time-out occurs since WAIT signal is input more than 80 μS in the user MPU.

Table 8-2 Operation State at Time-out

S	Operation state
0	HALT state
1 or 3	DMA state
2	First op-code fetch
5	SLEEP state
6	Second or third op-code fetch
7	CPU operation except op-code fetch

"S=x" represents operation state of the user MPU at time-out. (Refer to Table 8-2.) "A=xxxx xxxxx" represents the address where time-out occurs. Or a parity error occurs with emulation memory access.

(vi) **WRITE PROTECT ... Data is written to write protected area.

(vii) BLANK ... The user program is not executed.

(ii) to (v) indicates a user system abnormality and MPU stops. By pressing BREAK key, it goes into the ASE command input wait state. At command input wait state, the display disappears.

(b) CLOCK=x
6 ... 6.144MHz clock of the ASE is used as user MPU clock
U ... User system clock is used as user MPU clock.

(c) MODE=x
L ... Logical address input mode
P ... Physical address input mode

(d) SP=x
0 ... STOP condition isn't specified.
1 ... STOP condition is specified

(e) BR=&xxx
Displays the number of specified breakpoints.

(f) SYM=&xxxx
Displays the number of registered symbols.

(g) M=&xxxKB
Displays the size of remaining system memory.

(h) HOST=xxxxx
SD200 ..... Displays that it is a SD200 connect mode.
x x x x x ..... Displays that it is a general purpose host connect mode. Each character meaning is as follows:
①②③④⑤

ⓘ Displays baud rate
1... 19200BPS
2... 9600BPS
3... 4800BPS
4... 2400BPS
5... 1200BPS
6... 600BPS
7... 300BPS

② Displays data length
7 ... 7 bits 8 .... 8 bits
③ Displays parity
E ... Even parity
O ... Odd parity
D ... No parity
④ Displays stop bit length
1 ... 1-stop bit 2 ... 2-stop bit
⑤ Displays busy control method
Y ... Displays X-ON, X-OFF control
N ... Displays RTS, CTS control

(i) TR-BR-COND = x x x x
①②③④
Displays whether the trace condition or break condition is specified or not.

① Displays whether trace start condition is specified or not.
S ... Condition is specified.
Blank ... Condition isn't specified.
② Displays whether the trace save condition is specified or not.
R ... Condition is specified.
Blank ... Condition isn't specified.
③ Displays whether the break condition 1 is specified or not.
B1... Condition is specified.
Blank... Condition isn't specified.
④ Displays whether the break condition 2 is specified or not.
B2 ... Condition is specified.
Blank ... Condition isn't specified.

(j) COVERAGE=x
Displays the page number of coverage trace. Tracing access address in the range from $x0000 to $xFFFF is being captured. (x = 0,1,2,...,7)

(k) BASE=xx
Displays which base address is referred to in the logical address calculation. The first character shows the first priority base address and the second shows the second priority base address. Refer to section 6.2.2. Each character means as follows.
O ... Common area 0 base address
1 ... Common area 1 base address
B ... Bank base address

(l) INT=x
E ... Enables INT0 to INT2 interrupt input
D ... Disables INT0 to INT2 interrupt input

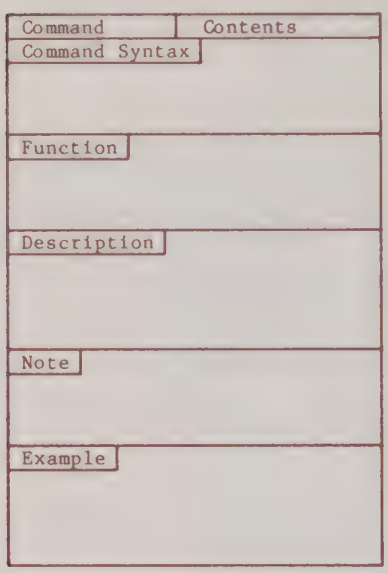
(m) NMI=x
E ... Enables NMI input
D ... Disables NMI input

(n) REF=x
E ... Enables refresh cycle trace
D ... Disables refresh cycle trace

9. ASE COMMANDS

This chapter provides the detailed description of ASE commands in the following format. The format is organized as shown below.

(1) The ASE commands are described in the format shown in Fig. 9-1.



Command: Command name. The abbreviation is enclosed by [ ].

Contents: Contents of the command

Command syntax: Describes the detailed syntax.

Function: Describes the function of the command.

Description: Describes how to use the command.

Note: Describes precautions when using the command.

Example: Describes the examples of the command usage.

(Note) Notes and examples are provided as required.

Fig. 9-1 Format of Command Description

Symbols used in the command description mean as follows.

[ ]: Parameter included in [ ] is ommittable.
{ }: { } means to select and specify one of several parameters included.
< >: The contents of < > is specified or displayed.
...: This means repetition of input just before .
△ : Spacing.(Used in the description of command format.)
(CR): Input of return key.

Note that the underlined part means to be input. (Omitted in the command syntax.)

9.1 ADDRESS — LOGICAL, AND PHYSICAL ADDRESS CALCULATION

Command Syntax

ADDRESS △ <value> [;{L / P}] CR

L : The value is regarded as a physical address, and calculates the logical address.
P : The value is regarded as a logical address, and calculates the physical address.

In the case of omission: Regards the value as the address which follows the current address mode, and calculates the other address.

Function

(1) The specified value is regarded as a physical address, the ASE calculates the logical address and indicates it.

(2) The specified value is regarded as a logical address, the ASE calculates the physical address and indicates it.

Description

(1) In the case of logical address calculation

```
:ADDRESS 11000;L CR
 LOGICAL  ADDRESS=1000
 PHYSICAL ADDRESS=11000
```

(2) In the case of physical address calculation

```
:ADDRESS 10;P CR
 LOGICAL  ADDRESS=10
 PHYSICAL ADDRESS=20010
```

(3) Interactive input format

```
:ADDRESS ? CR
 ADDRESS ? <address> CR
 GET LOGICAL ADDRESS OR PHYSICAL ADDRESS (L/P/return ) ? [ {L / P} ] CR
```

Notes

(1) Refer to section 6.2 for calculating an address.
(2) When the address mode to be specified is omitted, the ASE regards the specified value as the address following the current address mode and calculates the logical address or physical address from it. Input address mode is a physical address input mode at starting the ASE.

Example

(1) When calculates a logical address by input of the physical address;

```
:ADDRESS 11000;L (CR)
 LOGICAL  ADDRESS=1000
 PHYSICAL ADDRESS=11000
```

(2) When calculates a physical address by input of the logical address;

```
:ADDRESS 20;P (CR)
 LOGICAL  ADDRESS=0020
 PHYSICAL ADDRESS=11020
```

(3) When specifies the physical address depending on the input address mode (Interactive input format);

```
:ADDRESS ? (CR)
 ADDRESS ?11000 (CR)
 GET LOGICAL ADDRESS OR PHYSICAL ADDRESS (L/P/return) ?
 LOGICAL  ADDRESS=1000
 PHYSICAL ADDRESS=11000
```

9.2 ADDRESS_MODE SETTING AND DISPLAY OF INPUT ADDRESS MODE

Command Syntax

ADDRESS_MODE [△{L/P}] CR

L : Logical address input mode
P : Physical address input mode

Function

(1) This specifies whether an address is input as the logical address or physical address. (default:physical address)
(2) This indicates the current address input mode.

Description

(1) This specifies the address which is input with each command, it is used as either the logical address or the physical address. The specified address mode is valid until it is modified with the ADDRESS_MODE command.

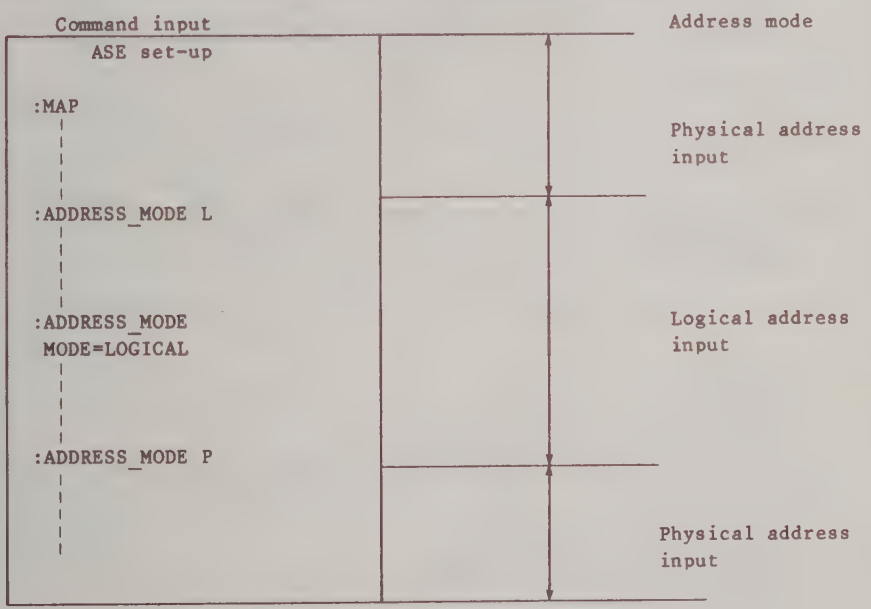


Fig. 9-2 Input Address Mode Setting

(2) In the case that the address mode is specified,

:ADDRESS_MODE {L/P} CR

L : Logical address
P : Physical address

(3) In the case that the current address mode is displayed.

```
:ADDRESS_MODE (CR)
MODE = <current address mode>
                    current address mode : LOGICAL
                                           PHYSICAL
```

(4) Interactive input format

(a) Setting

```
:ADDRESS_MODE ? (CR)
SET OR DISPLAY (S/D) ? S (CR)
LOGICAL OR PHYSICAL ADDRESS (L/P) ? [L] (CR)
                                    [P]
```

(b) Display

```
:ADDRESS_MODE ? (CR)
SET OR DISPLAY (S/D) ? D (CR)
MODE = <current address mode>
```

Example

(1) When specifies a logical address in the input address mode;

```
:ADDRESS_MODE L (CR)
```

(2) When displays the current input address mode;

```
:ADDRESS_MODE (CR)
MODE=LOGICAL
```

(3) Interactive input format

(a) Setting

```
:ADDRESS_MODE ? (CR)
SET OR DISPLAY (S/D) ?S (CR)
LOGICAL OR PHYSICAL ADDRESS (L/P) ?P (CR)
```

(b) Display

```
:ADDRESS_MODE ? (CR)
SET OR DISPLAY (S/D) ?D (CR)
MODE=PHYSICAL
```

9.3 A S S E M B L E

ASSEMBLE

Command Syntax

```
ASSEMBLE △ { <assembly statement> } CR
           {        ; C         }
```

C: Change memory mode

Function

(1) This converts the specified assembly statement into machine code, and displays it.
(2) This sets the machine code of the specified assembly statement from the specified address.

Description

(1) In the case that the machine code of the specified assembly statement is to be displayed.

```
:ASSEMBLE <assembly statement> CR
   < CODE >          <MNEM>          < OPRND >
 <machine code> <assembler mnemonics> <operand>
```

Display a machine code and its disassemble information.
Make sure that the correct assemble is done by this indication.
Refer to section 8.2 for the format of the assembly statement.

(2) When the machine code of the specified assembly statement is set to the specified address.

```
:ASSEMBLE ; C CR
ADDRESS ? <address> CR
KEY IN ASM STATEMENT
xxxx xxxxx ? <assembly statement> CR
   < CODE >          <MNEM>          < OPRND >
<machine code> <assembler mnemonics> <operand>
xxxx xxxxx ?  . CR
  ①   ②
  ① -- Logical address
  ② -- Physical address
```

After setting the result of assembly to the memory, the next address is displayed, and the ASE goes to command input wait state. To continuously revise the memory, input the assembly statement again. If it terminates, input a period(.).

(3) Interactive input format

(a) Machine code display

```
:ASSEMBLE ? CR
 CHANGE MEMORY DATA  (Y/N) ?   N CR
 ASSEMBLE STATEMENT ? <assembly statement> CR
      < CODE >          <MNEM>          < OPRND >
   <machine code> <assembler mnemonics> <operand>
```

(b) Memory change and machine code display

```
:ASSEMBLE ? CR
 CHANGE MEMORY DATA  (Y/N)  ?  Y CR
 ADDRESS ? <address> CR
 KEY IN ASM STATEMENT
 xxxx xxxxx ? <assembly statement> CR
    < CODE >          <MNEM>          < OPRND >
 <machine code> <assembler mnemonics> <operand>
 xxxx xxxxx ? . CR
```

Example

(1) When displays the machine code

```
:ASSEMBLE LD A,7F (CR)
 < CODE > <MNEM> < OPRND >
 3E7F       LD     A,7F
```

(2) When sets the machine code from $1000;

```
:ASSEMBLE;C (CR)
 ADDRESS ? 1000 (CR)
 KEY IN ASM STATEMENT
 1000 01000 ? LD HL,8000 (CR)
 < CODE > <MNEM> < OPRND >
 210080     LD     HL,8000
 1003 01003 ? JP 5C66 (CR)
 C3665C     JP     5C66
 1006 01006 ? . (CR)
```

(3) Interactive input format

(a) Machine code display

```
:ASSEMBLE ? (CR)
 CHANGE MEMORY DATA (Y/N) ?N (CR)
 ASSEMBLE STATEMENT ?JR NZ,FFF7 (CR)
 < CODE > <MNEM> < OPRND >
 20F5       JR     NZ,FFF7
```

(b) Change memory and machine code display

```
:ASSEMBLE ? (CR)
 CHANGE MEMORY DATA (Y/N) ?Y (CR)
 ADDRESS ? 2000 (CR)
 KEY IN ASM STATEMENT
 2000 02000 ? LD A,(EA54) (CR)
 < CODE > <MNEM> < OPRND >
 3A54EA     LD     A,(EA54)
 2003 02003 ? TST A (CR)
 ED3C       TST    A
 2005 02005 ? JP Z,B000 (CR)
 CA00B0     JP     Z,B000
 2008 02008 ? . (CR)
```

9.4 BACKUP BACKUP AND VERIFY OF ASE PROGRAM

Command Syntax

BACKUP[△V]CR

Function

(1) This makes a backup floppy disk for the proposed ASE program.
(2) This verifies the ASE program floppy disk.

Description

(1) Backup procedure

(a) Set a floppy disk formatted with FLOPPY_FORMAT command (refer to section 9.15) to the ASE station for backup.

(b) Key-in the BACKUP command.

```
:BACKUP CR'
 FILE-ID=<file data>
 BACKUP EXECUTION (Y/N) ? Y CR
```

When 'N' is input, the ASE goes into the command input wait state with no execution.

(c) Backup terminates when the system goes into the command input wait state.

(2) Verify procedure

(a) Set a floppy disk to be verified.

(b) Key-in the BACKUP command with specifying verify.

```
:BACKUP V CR
 FILE-ID=<file data>
 VERIFY EXECUTION (Y/N) ?  Y CR
```

When 'N' is input, the ASE goes into the command input wait state with no execution.

(c) When verify error occurs, the following message is output.

<ADDR>	<MEM>	<FILE>
xxxxxx	xx	xx
①	②	③

① Address ... the address at which an error occurs.
② Memory contents ....memory contents in the address where verify error occurs.
③ File contents....file contents in the address where verify error occurs.

Note

(1) The floppy disk should be formatted with FLOPPY_FORMAT command (refer to section 9.15) beforehand.
(2) FILE-ID after backup is the same as FILE-ID of the master floppy disk.

Example

(1) When backup the ASE program for the floppy disk formatted with FLOPPY_ FORMAT command;

```
:BACKUP (CR)
 FILE-ID=
 BACKUP EXECUTION (Y/N) ?Y(CR)
```

(2) When verifies the ASE program floppy disk;

```
:BACKUP V
 FILE-ID=HD64180 ASE (S180ASE1F) V1.0      COPYRIGHTED BY HITACHI,LTD. 1985
 VERIFY EXECUTION (Y/N) ?Y(CR)
```

9.5 BASE SETTING AND DISPLAY OF BASE TYPE FOR LOGICAL ADDRESS CALCULATION

Command Syntax

BASE [△<The first priority base type> [△<The second priority base type>]] ⓒR

Base type		
	0	---- Common area 0 base
	B	---- Bank area base
	1	---- Common area 1 base

Function

(1) In the case that the common area 0, the bank area and the common area 1 have the same address, this sets the area of the base address to be used as a base when the logical address is required.

(2) Displays the current base type.

Description

(1) As is shown in the following Fig. 9-3, in the HD64180, it is possible to have a maximum of 3 logical addresses by one physical address, depending on the MMU register setting. In this case, in order to determine which logical address is to be used, one of the common area 0, the bank area and the common area 1 is set with the order of priority.

(example)

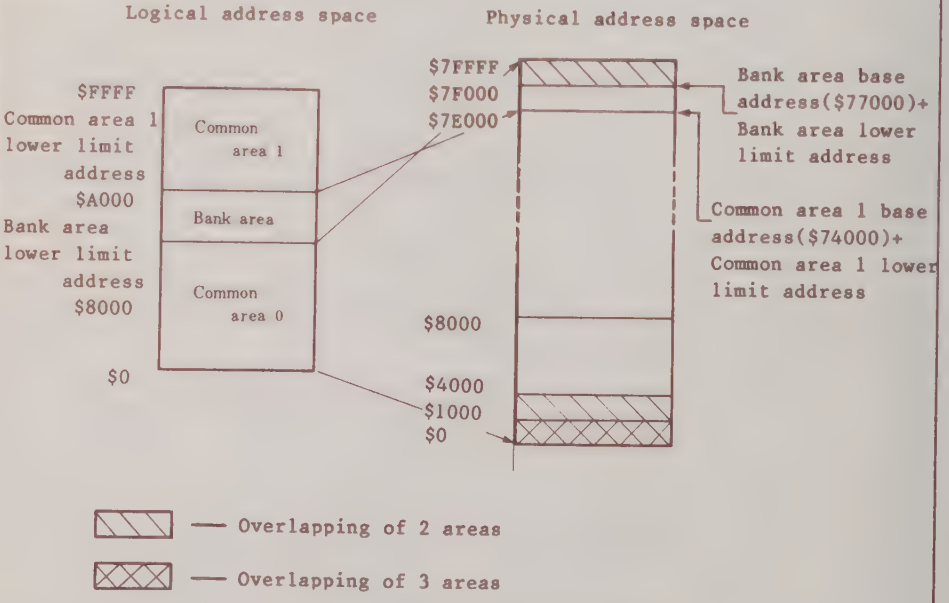


Fig. 9-3 Example having 3 logical addresses.

The logical address of the physical address $500.

① In the case that the common area 0 is the base type.
The logical address = $500

② In the case that the bank area is the base type.
The logical address = $500 - the bank area base address + $80000
= $9500

③ In the case that the common area 1 is the base type.
The logical address = $500 - the common area 1 base address + $80000
= $C500.

As for the MMU, if the physical address is $80000 or more, an $80000 subtracted address is used.
Refer to section 6.2.2 for calculating a logical address.

(2) Setting of the base type
: BASE <The first priority base type> [<The second priority base type>]CR

(a) The first and the second priority base type should not be the same.
(b) In initial state, the priority order is the following
The first priority base type 'O' --- Common 0 area
The second priority base type 'B' --- Bank area
(c) If the second priority base type is omitted, the base type is selected according to the next order.
i) Common area 0
ii) Bank area
iii) Common area 1

(3) Display of the base type in use.
:BASE CR
BASE1=<The first priority base type>
BASE2=<The second priority type>

The base type display	meaning
COMMON 0	The common area 0 is the base type
BANK	The bank area is the base type
COMMON 1	The common area 1 is the base type

(4) Interactive input format

a) Setting

```
:BASE ? CR
 SET OR DISPLAY (S/D) ? S CR
 FIRST  PRIORITY BASE IS COMMON0, BANK OR COMMON1 (0/B/1)  ? x CR
 SECOND PRIORITY BASE IS COMMON0, BANK OR COMMON1 (0/B/1)  ? x CR
```

b) Display

```
:BASE ? CR
 SET OR DISPLAY (S/D) ? D CR
 BASE1 = <The first priority base type>
 BASE2 = <The second priority base type>
```

Note

(1) When the common area 0, bank area, and common area 1 are not overlapped, because one physical address corresponds to one logical address, the priority of base does not have to be specified.

Example

(1) When displays the specified base type;

```
:BASE CR
 BASE1=COMMON0
 BASE2=BANK
```

(2) When calculates the address in the area with three logical address;

(a) Common area 0 as a base

```
:BASE 0 CR
:AD 500;L CR
 LOGICAL  ADDRESS=0500
 PHYSICAL ADDRESS=00500
```

(b) Bank area as a base

```
:BASE B CR
:AD 500;L CR
 LOGICAL  ADDRESS=9500
 PHYSICAL ADDRESS=00500
```

(c) Common area 1 as a base

```
:BASE 1 CR
:AD 500;L CR
 LOGICAL  ADDRESS=C500
 PHYSICAL ADDRESS=00500
```

(3) Interactive input format (setting and display)

```
:BASE ? CR
 SET OR DISPLAY (S/D) ?S CR
 FIRST  PRIORITY BASE IS COMMON0,BANK OR COMMON1 (0/B/1) ?0CR
 SECOND PRIORITY BASE IS COMMON0,BANK OR COMMON1 (0/B/1) ?BCR
:BASE ? CR
 SET OR DISPLAY (S/D) ?D CR
 BASE1=COMMON0
 BASE2=BANK
```

9.6 BREAK SETTING, DISPLAY AND DELETION OF BREAKPOINT FOR RAM AREA

Command Syntax

```
BREAK [ △{ <address>[;G]      } ]
      [  { [<address>];I      } ]   CR
      [  { -[<address>]       } ]
```

Function

(1) The breakpoint is set by changing the specified address instruction to the RST $30 instruction (in machine code $F7)
The GO command (refer to section 9.19) is terminated when this instruction is executed.

(2) If the G option is specified, the system interrupts execution with the specified address, displays the same information as the GO command termination, then continue the execution.

(3) Deletes the breakpoint.

(4) Displays all the set breakpoint. At this time, displays the pass counter value of the breakpoint and the execution time between breakpoints.

(5) If the I option is specified, the pass counter of the breakpoint and the execution time counter are cleared.

Description

(1) Sets the breakpoint.
:BREAK <address> CR

(2) Sets the breakpoint which interrupts execution and restarts execution.
:BREAK <address> ; G CR

(3) Deletes the breakpoint.
(i) In the case that the special breakpoint is deleted.
:BREAK - <address> CR

(ii) In the case that all the breakpoints are deleted.
:BREAK - CR

(4) Clears the pass counter of the breakpoint and the execution time counter.

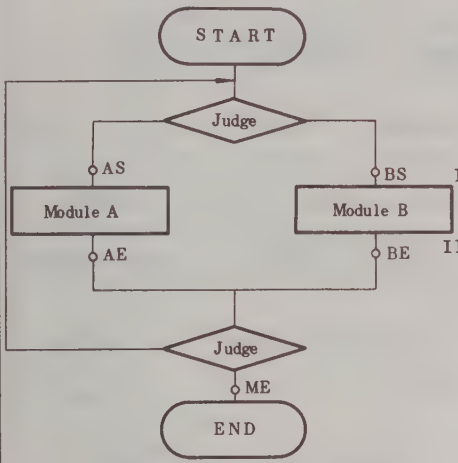
(a) Counter functions
The pass counter of the breakpoint and the execution time counter are used for analysis of the execution efficiency. Counting methods are as follows:

(i) The pass counter
1 point is added every time it passes the breakpoint.

(ii) The execution time counter
The execution time until it stops at the breakpoint, is added up.

(Ex.) Executing efficiency calculation

System Flowchart



<Procedure>

I. The user specifies a breakpoint at each address of AS, AE, BS, BE and ME in the program having the system flowchart shown left. The user specifies the G option at breakpoints except ME.

II. The user executes the program from the starting address with GO command.

III. After stopping at the breakpoint ME, the user gets information of all breakpoints with BREAK command.

IV. The user calculates the execution efficiency from the information of each breakpoint after stopping the breakpoint ME.

<Breakpoint>	<Pass Count of breakpoint>	<Passing time>	
AS	5	2000μS	
AE	5	5000μS	----- (Module A)
BS	10	4000μS	
BE	10	8000μS	----- (Module B)
ME	1	1000μS	
Total	31	20000μS	

Percentage of execution time of Module A in this system flowchart
5000/20000=25%

Percentage of execution time of Module B in this system flowchart
8000/20000=40%

Note) Passing count of breakpoints AS and BS is a total of execution time from START point or a breakpoint just before AS or BS.

(b) Pass counter clear

The count processing is done from the breakpoint setting time until it is deleted. There are two clears, the specified breakpoint counter clear and all breakpoints counter clear.

(i) Specified breakpoint counter clear

```
:BREAK <address> ; I CR
```

(ii) All breakpoints counter clear

```
:BREAK ; I CR
```

(5) All the breakpoints which are being set are displayed.

```
:BREAK CR
<  ADDR  >          <SYMBOL>   <OPT>   <PCNT>    <      RUN-TIME        >
xxxx xxxxx          xxxxxxxx     [G]    &xxxxx    &xxxxH:xxM:xxS:xxxxxxUS
 _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
  TOTAL PASS-COUNT = &xxxxxx
  TOTAL RUN-TIME   = &xxxxH:xxM:xxS:xxxxxxUS
```

① Address of breakpoint
② Symbol name of breakpoint (Displayed only when specified by symbol name)
③ Only when G option is specified, [G] is displayed.
④ Pass counter value of breakpoint (If the number exceeds 65535, "*****" is displayed.)
⑤ Execution time
⑥ Total of pass counter values of all breakpoints ("*****" is regarded as 65336.)
⑦ Total of execution time of all breakpoints

(6) Interactive input format

(a) Setting in the case that continuous execution is not specified.

```
:BREAK ? CR
DISPLAY BREAK POINTS (Y/N) ? N CR
SET, DELETE OR ALL DELETE (S/D/ALL) ? S CR
INITIALIZE OR CONTINUOUS GO (I/G/return) ? CR ... *1)
ADDRESS ? <address> CR
```

*1) Pressing CR specifies a breakpoint of the execution which is not continuous.

(b) In the case that the breakpoint of the continuous execution is set.
Input "G CR" by the option specification of the above (a).

(c) In the case that the particular breakpoint is deleted.

```
:BREAK ? CR
DISPLAY BREAK POINTS (Y/N) ? N CR
SET, DELETE OR ALL DELETE (S/D/ALL) ? D CR
ADDRESS ? <address> CR
```

(d) In the case that all breakpoints are deleted.

```
:BREAK ? CR
DISPLAY BREAK POINTS (Y/N) ? N CR
SET, DELETE OR ALL DELETE (S/D/ALL) ? ALL CR
```

(e) In the case that all breakpoints are indicated.

```
:BREAK ? CR
DISPLAY BREAK POINTS (Y/N) ? Y CR
```

(f) In the case that specified breakpoint counter is cleared.

```
:BREAK ? CR
DISPLAY BREAK POINTS (Y/N) ? N CR
SET, DELETE OR ALL DELETE (S/D/ALL) ? S CR
INITIALIZE OR CONTINUOUS GO (I/G/return)  ?  I  CR
ALL OR ONE BREAK POINT (ALL/addr)  ?  <address>  CR
```

(g) In the case that all the breakpoint counters are cleared.

```
:BREAK ? CR
DISPLAY BREAK POINTS (Y/N) ? N CR
SET, DELETE OR ALL DELETE (S/D/ALL) ? S CR
INITIALIZE OR CONTINUOUS GO (I/G/return)  ?  I  CR
ALL OR ONE BREAK POINT  (ALL/addr)  ?  ALL  CR
```

Notes

(1) A breakpoint can not be specified at logical address $2F. (The ASE cannot detect the breakpoint overlapped with an interrupt.) Be careful not to write $F7 in logical address $2F during user program execution. If the ASE executes $F7 (RST $30) of logical address $2F, it displays the following and stops. In this case, note that the stack contents is incorrect.

```
RST $30 INST. *INTERRUPT*
```

(2) Do not change the contents of the address specified as a breakpoint in the user program. Otherwise after stopping, the memory contents get recovered by the former data, so the memory contents will be incorrect.

(3) The address to be specified should be in the RAM area.

(4) The address to be specified should be the first op-code address of the instruction word. If any address except the first op-code address is specified, correct execution can't be done.

(Ex.) When the following 3-byte instruction is at $1000;

```
1000   FD 86 03        ADD   A, (IY + 3)
          ↑
```

If $1001 is specified, the instruction can't be executed correctly.

(5) If the user program has the RST $30 instruction, execution stops with the address. In this case, "RST $30 INST" is displayed as a condition of the GO command termination.
However, in the case that there are no breakpoint settings, the restart processing of the RST $30 instruction of the user program is executed without stop.

(6) The breakpoints which can be set are max. 255.

(7) The breakpoints which are specified by this command can be set, irrespective of the BREAK condition of the TRACE_BREAK_CONDITION command. (Refer to section 9.38.)

(8) If the program is executed in the trigger mode, the high-speed trace mode or the cycle reset mode, the breakpoint is invalid.
(The ASE does not set the RST $30 instruction.)

(9) In the TRACE_BREAK_CONDITION command, if <trace start condition> or <trace save condition> is specified, the BREAK condition is not satisfied. (The ASE does not set the RST $30 instruction.)

(10) The breakpoint holds a physical address in setting. Therefore, the current logical address to be displayed varies depending on the value of the MMU register.

(11) RST $30 instructions set to the memory just before executing GO command and after its execution the memory contents is recovered.

(12) When stopping at a breakpoint, the ASE uses two bytes of stack area since executing RST $30 instruction.

(13) When the user program execution stops at a breakpoint and restarts from the breakpoint, the ASE executes the instruction of the breakpoint by single-step and then continues real time emulation. Trace information by single-step execution is not saved.

(14) When passes the specified breakpoint with G option, the ASE clears the trace buffer once. Therefore, the trace information is saved only from the last breakpoint. Note that the trace information of instructions at the breakpoint is not saved.
Note that when an interrupt is input just at breakpoint, the user program stops even if G option is specified.

Example

(1) When specifies a breakpoint at $6809;

```
:BREAK 6809 CR
```

(2) When specifies a continuous breakpoint at $6800;

```
:BREAK 6800;G CR
```

(3) All breakpoints display

```
:BREAK CR
 <  ADDR  >  <SYMBOL><OPT><PCNT> <       RUN-TIME        >
 6809 06809                &1      &0000H:00M:00S:000021US
 6800 06800           [G] &1      &0000H:00M:00S:000032US
 ------------------------------------------------------------
 TOTAL PASS-COUNT = &2
 TOTAL RUN-TIME   = &0000H:00M:00S:000053US
```

(4) When cancels the specified breakpoint

```
:BREAK -6800 CR
```

(5) When cancels all breakpoints

```
:BREAK - CR
```

(6) When uses a symbol to specify an address;

```
:SYMBOL "SAMPLE=6000 CR
:BREAK "SAMPLE CR
```

(7) The specified pass counter of breakpoint clear (interactive input format);

```
:BREAK ? CR
 DISPLAY BREAK POINTS (Y/N) ?N CR
 SET,DELETE OR ALL DELETE (S/D/ALL) ?S CR
 INITIALIZE OR CONTINUOUS GO (I/G/return) ?I CR
 ALL OR ONE BREAK POINT (ALL/addr) ?"SAMPLE CR
```

9.7 BREAK_MASK — MASK SETTING AND DISPLAY OF BREAK CONDITION

Command Syntax

```
BREAK_MASK [△<mask specification>[△<mask specification>[△<mask specification>]]] (CR)
```

Function

(1) Among the BREAK conditions which are specified by the TRACE_BREAK_CONDITION command, the masks of the following items are specified.

.PC	program counter
MA	address
DATA	data

(2) Displays the mask which is specified.

Description

(1) The mask specification format.

(a) PC mask (0 to $7FFFF)
```
.PC = xxxxx
```
(b) address mask (0 to $7FFFF)
```
MA = xxxxx
```
(c) data mask (0 to $FF)
```
DATA = xx
```
(d) mask value

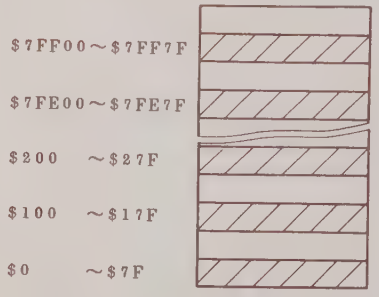
The mask value is specified to be 1 for a bit to be valid and 0 for a bit to be invalid.

(Ex.1) The user specifies BREAK_MASK to break when accessing 0 to $7F at one low-order byte of the address.

• Input command lines ---------The user specifies BREAK_MASK to check 2^7 bit of the address.

```
:BREAK_MASK MA=80 (CR)
:TRACE_BREAK_CONDITION (CR)
TRACE START CONDITIONS
? (CR)
TRACE RANGE CONDITIONS
? (CR)
BREAK CONDITIONS 1
? MA=0 (CR)
BREAK CONDITIONS 2
? (CR)
```

• Range of address to break



When accessing the shaded portion of the memory range shown left, the ASE breaks.

```
000 0000 0000 0000 1000 ($80)
                   ---Specified BREAK_MASK
000 0000 0000 0000 0000 ($0)
                   ---Break condition
        ↓  Logical multiply
xxx xxxx xxxx xxxx oxxx
   └→ Specified break condition
      "x" has nothing to do with
      reference of break condition.
```

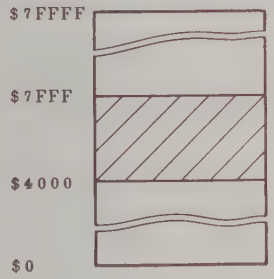
Fig. 9-4 Range of Address to Break (Ex.1)

(Ex.2) The user specifies BREAK_MASK to break when accessing from $4000 to $7FFFF. (The range is not specified with TRACE_BREAK_CONDITION because it is too large.)

• Input command lines

```
:BREAK_MASK  MA=7C000 (CR)      ---Specify BREAK_MASK as not to check 2^0 to
:TRACE_BREAK_CONDITION (CR)        2^13 bit of the address.
TRACE START CONDITIONS
? (CR)
TRACE RANGE CONDITIONS
? (CR)
BREAK CONDITIONS 1              ---Specify break condition as to break when
? MA=4000 (CR)                     accessing $4000 of memory address.
BREAK CONDITIONS 2
? (CR)
```

• Range of break




```
111 1100 0000 0000 0000 ($7C000)
                 ----Specified BREAK_MASK
000 0100 0000 0000 0000 ($4000)
                 ----Break condition
      ↓  Logical multiply
000 01xx xxxx xxxx
      └──→ Specified break condition
           "x" has nothing to do with reference
                of break condition.
```

When accessing the shaded portion of the memory range shown left, the ASE breaks.

Fig. 9-5 Range of Address to Break (Ex. 2)

(e) Initial state

All of the bits which are not mask-programmed, are valid, when started with the ASE.

.PC = 7FFFF MA = 7FFFF DATA = FF (The mask value during initial state)

(2) In the case that the using mask is indicated.

```
:BREAK_MASK (CR)
```

(3) Interactive input format

```
:BREAK_MASK ?(CR)
 SET OR DISPLAY (S/D) ? x (CR)
 PC MASK (0-7FFFF) ? xxxxx (CR)
 ADDRESS MASK (0-7FFFF) ? xxxxx (CR)
 DATA MASK (0-FF) ? xx (CR)
```

Notes

(1) If the mask is specified, all the mask specification should be done by MASK command at once. If the mask command is input twice, only the latter specification is valid. The omitted mask value will be regarded as the initial value.
(2) The mask is valid only for the break conditions 1 and 2 of TRACE_BREAK_CONDITION command.
(3) If the BREAK condition is a address range specified, mask operation is not executed.
(4) In the case of both a physical address input and a logical address input, the ASE uses the mask value specified with this command. As for a logical address input the ASE uses 2 low-order bytes of the mask value.

Example

(1) When displays the current mask;

```
:BREAK_MASK (CR)
 •PC=7FFFF   MA=7FFFF   DATA=FF
```

(2) When makes a low-order byte of the address bus valid;

```
:BREAK_MASK MA=FF (CR)
```

(3) When all the bits of program counter, twelve low-order bits of the address and all the bits of data are valid;

```
:BREAK_MASK ?  (CR)
 SET OR DISPLAY (S/D) ?S  (CR)
 PC MASK (0-7FFFF) ?7FFFF (CR)
 ADDRESS MASK (0-7FFFF) ?FFF (CR)
 DATA MASK (0-FF) ?FF (CR)
```

9.8 CHECK — PIN TEST OF MPU, MEMORY TEST

Command Syntax

CHECK[△<address 1>△<address 2>[△<write data>]] CR

address 1 --- starting address of memory test
address 2 --- end address of memory test
write data -- final write data.

Function

(1) Checks each pin of the HD64180 MPU. The check pins are as follows.

A0 to A18	WAIT	BUSREQ	BUSACK	RESET	NMI	INT0	INT1	INT2	
LIR	E	EXTAL	REF	IOE	ME	WR	RD	TEND1	DREQ1	ST	HALT

As for the signals of each pin, refer to the "HD64180 User's Manual".

(2) Tests the memory area of the specified address, and writes the specified data after testing.

Description

(1) Pin test of MPU
:CHECK CR

(a) The ASE checks address line by comparing the address set in MPU from the value output to the user system.
It checks $55555, $2AAAA, $7FFFF and $00000.
If there's something wrong, the following message will be displayed.

FAILED AT A0 - A18 SET ADDR <set address> ACCESS ADDR <access address>

(b) Except address line, the ASE checks if each signal is output according to each function.
If there's something wrong, the following message will be displayed.

FAILED AT <signal name>

(c) When there's no problem, the ASE goes into command input wait state without output. If the user system has not been connected yet, only EXTAL(clock input signal) causes an error.

FAILED AT EXTAL

(2) Memory test
:CHECK <address 1> <address 2> [<write data>] CR

The ASE tests the memory in the range of the specified address. The test procedure is shown on the next page. If omitted the write data, the ASE takes it $00.

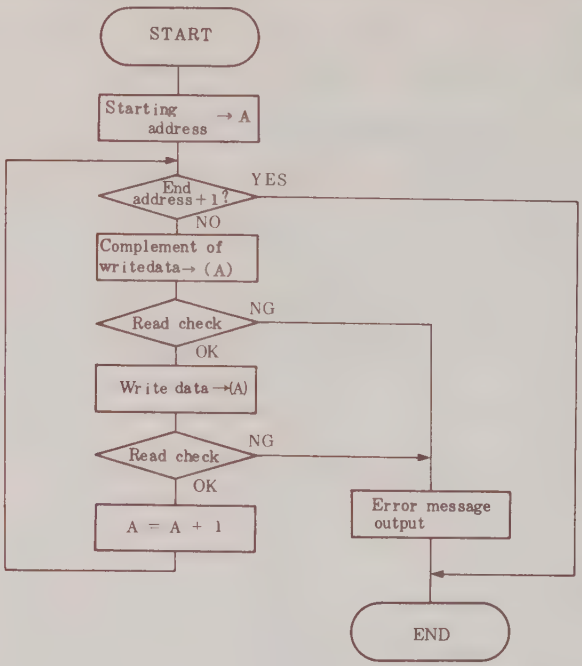


Fig. 9-6 Memory Test Procedure

If the error occurs, the following message will be displayed and the test will be finished.

FAILED AT <address> WRITE=<write data> READ=<read data>

(3) Interactive input format

(a) Pin test of MPU

```
:CHECK  ? CR
CHECK USER LINES OR MEMORY (L/M)  ? L CR
```

(b) Memory test

```
:CHECK  ? CR
CHECK USER LINES OR MEMORY (L/M)  ? M CR
TOP ADDRESS  ?  <address> CR
END ADDRESS  ?  <address> CR
WRITE DATA  ?  <data> CR
```

Note

(1) When specifying the memory test area by a logical address, a physical address in the area should be continuous. (Refer to section 6.2.4)

Example

(1) MPU pin check when the external clock is not input;

```
:CHECK (CR)
 FAILED AT EXTAL
```

(2) Memory test from $4000 to $5FFF (final write data $00);

```
:CHECK 4000 5FFF(CR)
```

(3) When test the memory from $6000 to $7FFF to take the final write data $FF (Interactive input format);

```
:CHECK ? (CR)
 CHECK USER LINES OR MEMORY (L/M) ?M (CR)
 TOP ADDRESS ?6000 (CR)
 END ADDRESS ?7FFF (CR)
 WRITE DATA ?FF (CR)
```

9.9 CLOCK — SELECTION AND DISPLAY OF CLOCK

Command Syntax

```
CLOCK [ △ { 6 } ] CR
      [   { U } ]
```

Function

(1) This selects whether the clock used in the real time emulation is input from the user, or the clock of 6.144MHz in the ASE is used.
(2) Displays the using clock type.

Description

(1) In the case that the clock of the user system is used.

```
:CLOCK U CR
```

(2) In the case that the ASE clock of 6.144MHz is used.

```
:CLOCK 6 CR
```

(3) In the case that the using clock type is indicated.

```
:CLOCK CR
 CLOCK = <type>
```

type: 6; 6.144MHz ASE clock is being used.
USER; The clock of the user system is being used.

(4) Initial status
When the ASE is set up, this checks whether the clock of the user system is input or not. If it is, it uses the clock of user system. In the case that the user clock is not being input, 6.144MHz clock in ASE is used.

(5) Interactive input format

(a) Display

```
:CLOCK ? CR
 SET OR DISPLAY (S/D) ? D CR
 CLOCK = <type>
```

(b) Setting

```
:CLOCK ? CR
 SET OR DISPLAY (S/D) ? S CR
 CLOCK KIND (U/6) ? { 6 } CR
                    { U }
```

Notes

(1) When a user system is connected, use the user clock as much as possible. If using the ASE clock, which does not match with the clock on the user system, the ASE might not operate normally.

(2) The clock is modified, input the RESET signal into HD64180. Therefore, each register contents will be the same as that at reset. Refer to "HD64180 User's Manual" for details.

(3) When using user clock, powering the user system OFF causes to stop inputting clock to the ASE and terminate the ASE program. Then the ASE goes into the initial state, displaying "USER SYSTEM NOT READY".

Example

(1) When uses CLOCK in the user system;

```
:CLOCK U CR
 *** RESET IN BY ASE !
```

(2) When uses CLOCK 6.144MHz in the ASE;

```
:CLOCK 6 CR
 *** RESET IN BY ASE !
```

(3) When uses CLOCK 6.144MHz in the ASE (interactve input format);

```
:CLOCK ? CR
 SET OR DISPLAY (S/D) ?S CR
 CLOCK KIND (U/6) ?6 CR
 *** RESET IN BY ASE !
```

(4) When displays the current CLOCK type;

```
:CLOCK CR
 CLOCK=6.144MHZ
```

9.10 COMMAND_CHAIN — INPUT ASE COMMAND FROM FILE

Command Syntax

COMMAND_CHAIN <file name> CR

Function

(1) This inputs the ASE command sequentially from the specified file, and executes it. (only with SD200)

Description

(1) The ASE command is input one by one from the specified file, and executes it sequentially. Use "CP/M-68K CRT Editor" to make the command file.

```
:COMMAND_CHAIN <file name> CR
```

```
Executes the file contents in
the right sequentially.
```

Contents of the command file

```
MAP 0 1FFF = UR
GO  50
TRACE
```

(2) Interactive input format

```
:COMMAND_CHAIN ? CR
 FILE NAME ? <file name> CR
```

Note

(1) Don't include this command in the file.
(2) Only the SD200 CP/M-68K system is used as a host system.
(3) Inputting BREAK key ends inputting commands from the command file. Inputting BREAK key during GO command execution is regarded as stop requirement with GO command.

Example

(1) When executes the command file "SAMPLE.TSET" in SD200 disk drive B;

```
:COMMAND_CHAIN B:SAMPLE.TST CR
```

(2) Interactive input format of the above (1);

```
:COMMAND_CHAIN ?  CR
 FILE NAME ?B:SAMPLE.TST CR
```

9.11 CONVERT DATA CONVERSION

Command Syntax

CONVERT △<data> CR

Function

(1) This converts the result of the data into hexadecimal, decimal, octal, binary number and ASCII codes.

Description

(1) The following are used for the display.
<in hexadecimal> <in decimal> <in octal> <in binary> <in ASCII code>

(2) Interactive input format
:CONVERT ? CR
DATA ? <data> CR

Notes

(1) All of the input data and the results are regarded as 4-byte data.
(2) If the corresponding ASCII code does not exist, period "." is displayed.
(3) The constant (including the expression) written in "8.1 Command Syntax" can be specified as data.

Example

(1) When the input data is in hexadecimal;

```
:CONVERT 7F (CR)
 7F &127 @177 %01111111 '....'
```

(2) When the input data is in decimal;

```
:CONVERT &100 (CR)
 64 &100 @144 %01100100 '...d'
```

(3) When the input data is in octal;

```
:CONVERT @77 (CR)
 3F &63 @77 %00111111 '...?'
```

(4) When the input data is in binary;

```
:CONVERT %10100101 (CR)
 A5 &165 @245 %10100101 '....'
```

(5) When the input data is in characters;

```
:CONVERT 'ABCD'(CR)
41424344 &1094861636 @10120441504 %01000001010000100100001101000100 'ABCD'
```

(6) When the input data expression is negative (Interactive input data);

```
:CONVERT ?(CR)
DATA ?0-1(CR)
FFFFFFFF 3429496729S @37777777777 %11111111111111111111111111111111 '....'
```

(7) When the input data consists of characters and numbers (Interactive input format);

```
:CONVERT ? (CR)
DATA ?'A'+@17 (CR)
 50 &80 @120 %01010000 '...P'
```

9.12 COVERAGE — INITIALIZATION AND RESULTS DISPLAY OF COVERAGE TRACE

Command Syntax

(1) Initialization

COVERAGE △ I[=<page number>] ⓒR

page number 0 to 7
(default value = 0)

(2) Results display

COVERAGE [△<address 1>△<address 2>] [;{D / A}] ⓒR (A, if an option omitted.)

address 1 : display starting address
address 2 : display end address
D : data are displayed in DUMP list
A : data are displayed in address value

Function

(1) Capturing the accessed address during the user program execution. Cover range is a 64KB unit.

(2) Tracing data which is captured is displayed by an address value or dump list.

(3) Clear the data, and initialize the range to be captured.

Description

(1) Usage method

Example of a command	Contents
:COVERAGE I = 0	COVERAGE buffer initialization Access address range to be captured: $0 to $FFFF
:GO	
:GO	
:COVERAGE	Display of reference or modified address after COVERAGE buffer initialization
:COVERAGE I = 1	COVERAGE buffer clear Access address range to be captured: $10000 to $1FFFFF

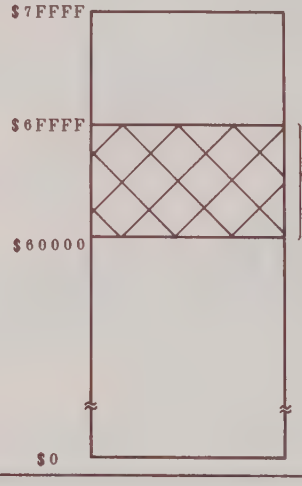
Fig. 9-7 Example of COVERAGE Command

(2) In the case of initialization
:COVERAGE I [=<page number>] (CR) (Page number is 0, if omitted.)

Only the 64KB range can be covered for address trace data. Therefore, which range of physical address can be captured is specified by page number.

Page number	Trace cover range (physical address)
0	$0 to $FFFF
1	$10000 to $1FFFF
2	$20000 to $2FFFF
3	$30000 to $3FFFF
4	$40000 to $4FFFF
5	$50000 to $5FFFF
6	$60000 to $6FFFF
7	$70000 to $7FFFF

(Ex.)



If the page number is 6, only this address range of accessed address can be traced.

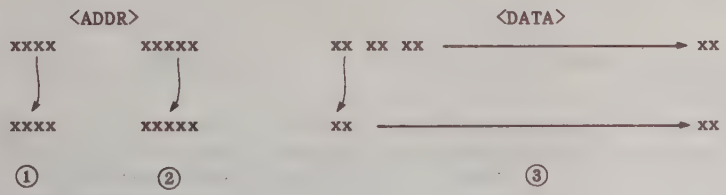
Fig. 9-8 Cover Range of COVERAGE Information

(3) In the case of displaying the access address data

(a) Dump list display

:COVERAGE [<address 1> <address 2>] ; D (CR)

Coverage data from <address 1> to <address 2> are displayed. Without specifying the addresses, all captured data on the specified page are displayed.
Display format is shown as follows.



① Logical address
② Physical address
③ Access data

The data is displayed in hexadecimal(0 to FF). Each address corresponds to each bit. An address which is referred or modified is displayed as 1.

(Ex.)

< ADDR >		<DATA>				
0080	10080	8F	00	00	00	00

Starting data, $8F shows that reference and modification of $10080, $10084, $10085, $10086, $10087 are executed.

Data values	(8)	(F)
Meaning of a bit	1 0 0 0	1 1 1 1
	↑	↑
	$10080	$10084

(b) Display by an address value

:COVERAGE [<address 1> <address 2>] ;A CR

Coverage data from <address 1> to <address 2> are displayed. Without specifying the addresses, all captured data on the specified page are displayed.
Display format is as follows.

```
    <ADDR>      <ADDR>          <ADDR>
xxxx    xxxxx[-]xxxx xxxxx[-]xxxx  xxxxx ----------------
 ↓                    ↓                ↓
xxxx    xxxxx[-]xxxx xxxxx[-]xxxx  xxxxx ----------------
①         ②  ③
```

① Logical address	Logical addresses referred or modified are displayed.
② Physical address	Physical addresses referred or modified are displayed.
③ -	If - appears, it shows that all addresses in the range from just before this address to just after this address are accessed.

(4) Interactive input format

(a) Initialization

```
:COVERAGE ? CR
INITIALIZE OR DISPLAY (I/D) ? I CR
PAGE NO (0-7) ? <page number> CR
```

(b) Display of trace data

```
:COVERAGE ? CR
INITIALIZE OR DISPLAY (I/D) ? D CR

DUMP OR ADDRESS IMAGE (D/A) ? {D/A} CR
DISPLAY ALL RANGE (Y/N) ? N CR
TOP ADDRESS ? <address> CR
END ADDRESS ? <address> CR
```

Note

(1) With the ASE start-up COVERAGE capturing page number is 0.
(2) When you specify the display range, specify a multiple of 8 (including 0) for starting address and (a multiple of 8)-1 for end address. Otherwise, starting address is cut down and end address is raised.
(3) Except addresses accessed by the user program, addresses accessed by the ASE are displayed as COVERAGE information.
 (a) BREAK key input during SLEEP or WAIT (Use SP or SP-1 by NMI input.)
 (b) Breakpoint execution (Use SP or SP-1 by NMI input)
 (c) GO or STEP command operation (Set only COVERAGE information of PC+1 when the program stops.)

Example

(1) When displaying access address information by address value display;

```
:COVERAGE 0 3FF (CR)
<  ADDR  >   <  ADDR  >   <  ADDR  >   <  ADDR  >   <  ADDR  >
0000 00000 - 0102 00102   0140 00140 - 0142 00142   0170 00170 -
0172 00172   0182 00182 - 0184 00184   01CC 001CC - 01CE 001CE
0240 00240 - 03FF 003FF
```

(2) When specifying the dump image display;

```
:COVERAGE 0 3FF;D (CR)
<  ADDR  >   <                          DATA                          >
0000 00000   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0080 00080   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0100 00100   E0 00 00 00 00 00 00 00  E0 00 00 00 00 00 E0 00
0180 00180   38 00 00 00 00 00 00 00  00 0E 00 00 00 00 00 00
0200 00200   00 00 00 00 00 00 00 00  FF FF FF FF FF FF FF FF
0280 00280   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0300 00300   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
0380 00380   FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF
```

(3) When initializing with page 0 (Interactive input format);

```
:COVERAGE ? (CR)
 INITIALIZE OR DISPLAY (I/D) ?I (CR)
 PAGE NO (0-7) ?0 (CR)
```

9.13 DISASSEMBLE — DISASSEMBLE OF MEMORY CONTENTS

Command Syntax

DISASSEMBLE△<starting address> [△<end address>] CR

Function

(1) Memory contents of a specified address are disassembled and the assembly mnemonic is displayed.

(2) When an end address is not specified, only 16 assembly mnemonics are displayed.

Description

(1) Memory contents from a specified starting address to an end address are displayed by the following format.

```
:DISASSEMBLE  <starting address>  <end address>  CR
<  ADDR  >      <  CODE  >       <MNEM>   <OPRND>
   (1)            (2)            (3)      (4)

XXXX  XXXX      XXXXXXXX         XXXXX    XXX  XX
```

(1) Address
(2) Machine code
(3) Assembly mnemonic
(4) Operand

(2) In the case of displaying only 16 assembly mnemonics

```
:DISASSEMBLE  <starting address> CR
```

(3) Interactive input format

```
:DISASSEMBLE    ? CR
TOP ADDRESS ?   <starting address> CR
END ADDRESS ?   <end address>  CR
```

Note

If there is not above-mentioned assembly mnemonic, the data are displayed as "DC xx" and the next instruction is regarded to be at the address one byte forward.

Example

(1) When disassembling from $1000 to $100B;

```
:DISASSEMBLE 1000 100B (CR)
 <  ADDR  >  < CODE >  <MNEM>  < OPRND >
1000 01000  210011      LD      HL,1100
1003 01003  3EC9        LD      A,C9
1005 01005  0600        LD      B,00
1007 01007  77          LD      (HL),A
1008 01008  23          INC     HL
1009 01009  10FC        DJNZ    1007
100B 0100B  C9          RET
```

(2) Interactive input format of example (1);

```
:DISASSEMBLE ? (CR)
TOP ADDRESS ?1000 (CR)
END ADDRESS ?100B (CR)
 <  ADDR  >  < CODE >  <MNEM>  < OPRND >
1000 01000  210011      LD      HL,1100
1003 01003  3EC9        LD      A,C9
1005 01005  0600        LD      B,00
1007 01007  77          LD      (HL),A
1008 01008  23          INC     HL
1009 01009  10FC        DJNZ    1007
100B 0100B  C9          RET
```

9.14 DISPLAY — SET AND DISPLAY CONSOLE DISPLAY MODE

Command Syntax

DISPLAY [△{N / S / P}] ㊀

N : general scroll mode
S : full screen display scroll mode
P : full screen display page mode

Function

(1) Set display mode of the console.
(2) Current display mode is indicated on the screen.

Description

(1) The following describes each display mode.
  (a) General scroll mode
      The key input data and execution results of each command are output successively. Any other consoles can be used as well as the SD200 console. Output characters are only ASCII characters, carriage return code ($0D), line feed code ($0A), and backspace code ($08), so any console may be used if they accept these codes. Refer to section 4.2 for details of connection.
      Only this mode is available when using any console other than the SD200.

(b) Full screen display scroll mode

Only the SD200's system console can specify this mode. Display format is shown as follows.

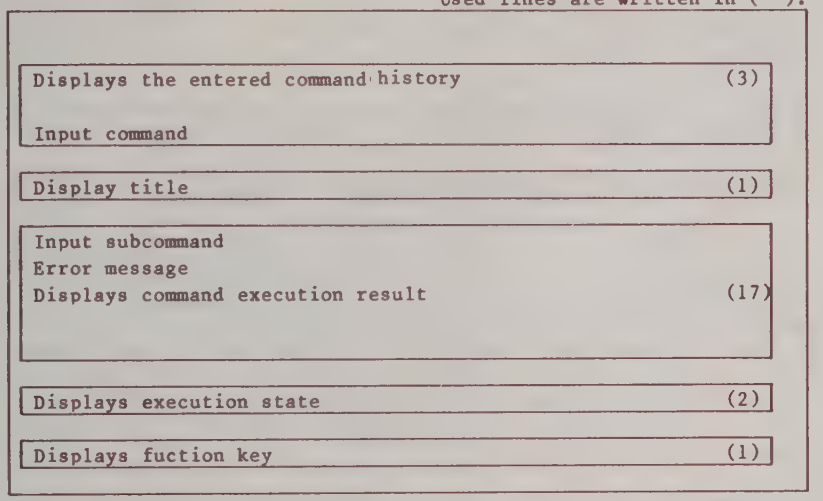


Fig. 9-9 Display Format in Full Screen Display Mode

Refer to "8.4 Full Screen Display Mode" for details.

As to the scroll mode, when command execution result are displayed in more than 17 lines, the screen scrolls up within this execution result display area. Top most line is removed, the screen scrolls up by one line and a new line is displayed at the bottom.

(c) Full screen display page mode

Only the SD200 console can specify this mode. Display format is the same as that of (b).

As to the page mode, if command execution results are displayed in more than 17 lines, display stops when the 17th line is displayed. When the (CR) key is pressed, the next 17 lines are displayed

(2) Specification of the general console mode
: DISPLAY N CR

(3) Specification of the screen display scroll mode
: DISPLAY S CR

(4) Specification of the screen display page mode
: DISPLAY P CR

(5) Display of the current mode
: DISPLAY CR

In this case, the following messages are displayed.

Display	Meaning
NORMAL MODE (N)	General mode
PAGE EDIT LINE SCROLL MODE (S)	Screen editing scroll mode
PAGE EDIT PAGE SCROLL MODE (P)	Screen editing page mode

(6) Interactive input format

(a) Display mode set
: DISPALY ? CR
SET OR DISPLAY (S/D)? S CR
DISPLAY MODE (N/S/D)? x CR

(b) Mode display
: DISPALY ? CR
SET OR DISPLAY (S/D)? D CR
<current mode>

Note

(1) When console any other than the SD200 is connected, and if the scroll mode or the page mode is specified, displayed contents can not be guaranteed.

(2) As for the connection of console, refer to 4.2.

Example

(1) Set and dislay of full general scroll mode ;

```
:DISPLAY N CR
:DISPLAY CR
 NORMAL MODE(N)
```

(2) Set and display of screen display scroll mode (Interactive input format);

```
:DISPLAY ? CR
 SET OR DISPLAY (S/D) ?S CR
 DISPLAY MODE (N/S/P) ?S CR
:DISPLAY CR
 PAGE EDIT LINE SCROLL MODE(S)
```

(3) Set and display of screen display page mode (Interactive input format);

```
:DISPLAY ? CR
 SET OR DISPLAY (S/D) ?S CR
 DISPLAY MODE (N/S/P) ?P CR
:DISPLAY ? CR
 SET OR DISPLAY (S/D) ?D CR
 PAGE EDIT PAGE SCROLL MODE(P)
```

9.15 FLOPPY_FORMAT — A FLOPPY DISK FORMATTING

Command Syntax

```
FLOPPY_FORMAT (CR)
```

Function

(1) Formatting the 3.5-inch floppy disk

Description

(1) Formatting the 3.5-inch floppy disk set in ASE

```
:FLOPPY_FORMAT (CR)
FILE-ID = <file data>
EXECUTION (Y/N) ? Y (CR)
```

- When "N" is input, the ASE goes into the command input wait state with no execution.
- A file data display is the former one, but if the floppy disk is not formatted yet, the file data will be displayed as follows.

```
*** UNFORMATTED FD ***
```

(2) Interactive input format
It is the same as single-line input.

Notes

(1) After formatting, the file data is blank.

Example

(1) When formatting unformatted floppy disk;

```
:FLOPPY_FORMAT (CR)
FILE-ID=*** UNFORMATTED FD ***
EXECUTION (Y/N) ?Y (CR)
```

9.16 FLOPPY_LOAD — LOAD FROM A FLOPPY DISK

Command Syntax

```
FLOPPY_LOAD (CR)
```

Function

(1) The contents of a floppy disk are loaded into the user memory.

Description

(1) The contents of 3.5-inch floppy disk in the ASE are loaded into the user memory.

The contents to load is the same contents saved by FLOPPY_SAVE command (refer to section 9.17).

```
: FLOPPY_LOAD (CR)
  FILE - ID = <file data>
  EXECUTION (Y/N) ? Y (CR)
  TOP ADDRESS : <starting address>
  END ADDRESS : <end address>

  <file data> ---------- file data set by FLOPPY_SAVE command
  <starting address> ---- loaded starting address
  <end address> --------- loaded end address
```

If "N" is input, the ASE becomes the next command wait state without any execution.

(2) Interactive input format

It is the same as single-line input.

Example

(1) When loading the contents of floppy disk into the user memory;

```
:FLOPPY_LOAD (CR)
 FILE-ID=SAMPLE1
 EXECUTION(Y/N) ?Y (CR)
 TOP ADDRESS=0000 00000
 END ADDRESS=0FFF 00FFF
```

9.17 FLOPPY_SAVE — SAVE USER MEMORY CONTENTS INTO A FLOPPY DISK

Command Syntax

FLOPPY_SAVE△<address 1>△<address 2> (CR)

Function

(1) User memory contents of a specified address range is saved into floppy disk.

Description

(1) User memory contents of a specified address range is saved into a 3.5-inch floppy disk set in ASE.

```
:FLOPPY_SAVE <address 1>  <address 2>  (CR)
 FILE-ID = <former file data>
 EXECUTION (Y/N)  ?  Y (CR)
 FILE-ID = <new file data> (CR)
```

Memory contents of from <address 1> to <address 2>are saved into a floppy disk. Before saving, <former file data> is displayed, so confirming this data, input "Y". If "N" is input, the ASE goes to the next command wait state without any execution. If "Y" is input, respond requirement is displayed to input new <file data>, so set it. Any character strings can be input to the file data. The maximum character number is 70, and the character exceeding 70 is ignored.

(2) Interactive input format

```
: FLOPPY_SAVE ? (CR)
  TOP ADDRESS ? <address 1> (CR)
  END ADDRESS ? <address 2> (CR)
  FILE-ID = <former file data>
  EXECUTION (Y/N)  ?  Y (CR)
  FILE-ID = <new file data> (CR)
```

Note

(1) A floppy disk to be used should be formatted by FLOPPY_FORMAT command (refer to section 9.15) beforehand.
(2) When specifying the area to be saved by a logical address, the physical address in the area should be continuous. (Refer to section 6.2.4.)
(3) In this command, one floppy disk is always used for one file. Therefore, to make several files, as many 3.5-inch floppy disks should be provided.

Example

(1) When saves the contents of the user memory from $0 to $FFF into the floppy disk (Interactive input format);

```
:FLOPPY_SAVE ? CR
 TOP ADDRESS ?0000 CR
 END ADDRESS ?0FFF CR
 FILE-ID=FDTEST
 EXECUTION (Y/N) ?Y CR
 FILE-ID ?SAMPLE1 CR
```

9.18 FLOPPY_VERIFY USER MEMORY CONTENTS AND FLOPPY DISK VERIFY

Command Syntax

```
FLOPPY_VERIFY (CR)
```

Function

(1) Contents of a floppy disk and a user memory are verified. If they are different, error message is displayed.

Description

(1) Contents of a 3.5-inch floppy disk and a user memory are verified.
The contents verified are the same as the ones saved by FLOPPY_SAVE command (refer to section 9.17). That is, an address range verified is the address range saved.

```
: FLOPPY_VERIFY (CR) ?
  FILE-ID = <file information>

  EXECUTION (Y/N)  ?  Y (CR)
  TOP ADDRESS : <starting address>
  END ADDRESS : <end address>

   <file data> ------------ file data set by FLOPPY_SAVE command
   <starting address> ----- verified starting address
   <end address> ---------- verified end address
```

If "N" is input, the ASE becomes the next command wait state. If there is a verify error, the following message is displayed.

```
 < ADDR >        <FILE>       <MEM>
xxxx xxxxx         xx           xx
 (1)   (2)         (3)          (4)
```

(1) Logical address
(2) Physical address
(3) File contents — File contents of an address in which verify error
(4) Memory contents — Memory contents of an address in which verify error

(2) Interactive input format
It is same as single-line input.

Example

(1) When verifies the contents of the floppy disk with that of the user memory;

```
:FLOPPY_VERIFY  (CR)
 FILE-ID=SAMPLE1
 EXECUTION (Y/N) ?Y (CR)
 TOP ADDRESS=0000 00000
 END ADDRESS=0FFF 00FFF
```

Command Syntax

GO [A[<starting address>][;[N]] {T / H / R = <n>}]] CR

starting address : RESET, address
T : trigger mode
H : high-speed trace mode
R = <n> : cycle reset mode, n = 1 to 6
N : execution begins from NMI interrupt servicing state (NT, NH, NR = <n>)

Function

(1) User program is executed from a specified address in real time.
(2) Executes the user program with saving bus data in bus cycle unit or 125nS unit: (high-speed trace mode).
(3) At a break condition, a trigger signal for the oscilloscope is output from the TRIG output pin. (trigger mode)
(4) When a reset signal is input compulsory every certain time during execution, a trigger signal for the oscilloscope is output from the TRIG output pin. (cycle reset mode)

Description

(1) User program from a specified address is executed in real time.
Execution is performed until the following conditions are satisfied.

(a) Break condition set by TRACE_BREAK_CONDITION command (refer to section 9.38) is satisfied.(A trigger mode, and cycle reset mode are excluded.)
(b) When the program execution reaches to a breakpoint specified by a BREAK command and an option of a continuous execution is not specified to the breakpoint.
(c) A BREAK key is input.
(d) Trace start condition is specified and a trace buffer is fulled after a trace starts.

(2) During the execution of a user program, the data is saved in a trace buffer by the trace start condition or trace save condition which is set by the TRACE_BREAK_CONDITION command. 2048-cycle data can be saved in the trace buffer. If data exceeds the limit, the earlier data area is overwritten, so that the last 2048-cycle data are always buffered. (If the execution is less than 2048 cycles, only data for executed cycles are buffered.)

1 cycle of this trace saving is as follows.

(a) Machine cycle unit (bus cycle trace mode)
(b) 125 nS unit (high-speed trace mode)

In high-speed trace mode, specify "H" or "NH" as an option as follows. Otherwise, trace data is saved by a bus cycle unit.

: GO [<starting address>] ;{H / NH} CR

(3) In the trigger mode, when TRACE_BREAK_CONDITION is effective, execution does not stop and a trigger signal for the oscilloscope is output.

: GO [<starting address>] ;{T / NT} CR

(4) In the cycle reset mode, a reset signal is input forcedly to HD64180 at each specified time. At the same time a trigger signal to the oscilloscope is output, and execution is continued.
In this mode, a break condition is ignored.

: GO [<starting address>] ;{R / NR} = n CR

value of n	repeated time
1	20 µS
2	100 µS
3	1 mS
4	10 mS
5	100 mS
6	655.35 mS

(5) The following specification is possible as an execution starting address.

: GO△{ { RESET / <address> } } [; <option>] ㊥

RESET : After a RESET signal is input to HD64180 execution begins from a restart address.
address : Execution begins from a specified address
no specification : Execution begins from an address which the current program counter

(6) N option specification

If a user program execution stops, when a user system is in NMI process (IEF1=0, IEF2=1), N option should be specified whenever the execution is continued. If continuous execution is performed without specification, IEF1 forcedly becomes 1 by the ASE, so there is a possibility that execution is not normally performed.

Only flag of IEF2 (IF=x) is displayed in a register, so judgement whether it is in NMI operation or not is performed by the user. Refer to "HD64180 User's Manual" for details of IEF1 and IEF2 (Interrupt Enable Flag).

(7) When GO command is input, the following message is displayed before user program execution;

PROCESS STARTED FROM xxxx ① xxxxx ②

① Execution starting logical address
② Execution starting physical address

(8) When GO command ends, the following message is displayed.

①

② RUN - TIME = &xxxxH : xxM : xxS : xxxxxxUS
③ <The reason for stopping>

① The current contents of each register.
② Executed time of the user program. It is displayed in decimal and possible measuring time is about 1 year (2^{45} μS). The relative accuracy is ± 2μS.

③ -------- The reason for stopping the GO command is displayed. Display contents are shown as follows.

Display	Meaning
(a) WRITE PROTECT	The data is written into write protect area.
(b) RST $30 INST	RST $30 instruction except at breakpoint is executed.
(c) BREAK CONDITION 1	A break condition 1 is satisfied.
(d) BREAK CONDITION 2	A break condition 2 is satisfied.
(e) TRACE BUF FULL	A trace start condition is specified and trace data of 2048 cycles is saved.
(f) BREAK KEY	The break key is input.
(g) BREAK POINT	Execution stops by reaching the breakpoint specified by BREAK command. If there is a continuous specification, execution of the user program restarts.
(h) BREAK POINT *INTERRUPT*	Interrupt is input during RST $30 instruction execution at breakpoint. PC shows interrupt servicing address and return address in the stack area shows the breakpoint.
(i) RST $30 INST *INTERRUPT*	Interrupt is input during RST $30 instruction execution except at breakpoint. PC shows interrupt servicing address and return address in the stack area shows RST $30 instruction.
(j) PARITY ERROR	A parity error occurs with emulation memory access.

(9) After starting the user program execution, the ASE displays the execution state. Display part on the screen is different whether using full screen display mode or not.

(a) Execution state display in full screen display mode
Execition state is displayed 21 lines in the part of user state display.

(b) Execution state display in the other mode
Displays from current cursor position.

(c) Display description

Display	Description
(i) ** RUN PC=xxxx xxxxx	Displays the program counter (first op-code fetch address) in operation. In the mode other than full screen display mode, displays only when restarting in the state of (ii) to (v).
(ii) ** RESET	RESET signal is input to user MPU.
(iii) ** SLEEP	SLEEP state
(iv) ** BUS REQUEST	BUS REQUEST signal is input to user MPU.
(v) ** TOUT S=x A=xxxx xxxxx	WAIT signal is input to user MPU. S=x represents the following operation state and A=xxxx xxxxx represents the address where time out occurs. Or a parity error occurs with emulation memory access.

Value of S	Operation state
0	HALT state
1 or 3	DMA operation
2	first op-code fetch
5	SLEEP state
6	second or third op-code fetch
7	CPU operation except op-code fetch

Note) In the case of (iv) to (v), pressing the BREAK key stops user program execution. At this time, the ASE stops by input of RESET signal to user MPU. Notice that the internal I/O register is reset because of this.

Modification of execution state display is performed in 300mS unit.

(10) Interactive input format

(a) In the case of execution from the specified address

```
:GO  ?  CR
START ADDRESS (RESET/addr/return)  ?  1000  CR
MODE HIGH SPEED (H), TRIGGER (T), CYCLE RESET (R), FROM NMI (N)
(H/T/R/N/NH/NT/NR/return)  ?  CR
```

(b) In the case of execution in cycle reset mode (10mS) from the current address

```
:GO  ?  CR
START ADDRESS (RESET/addr/return)  ?  CR
MODE HIGH SPEED (H), TRIGGER (T), CYCLE RESET (R), FROM NMI (N)
(H/T/R/N/NH/NT/NR/return)  ?  R  CR
CYCLE TIME 20U, 100U, 1M, 10M, 100M, 655MS (1-6)  ?  4  CR
```

Note

(1) Specify the stack pointer before inputting GO command or in the program to execute. The ASE uses 2-byte stack area in the following case.
   (a) When the user stops at the breakpoint
   (b) When the user specifies a break condition (TRACE_BREAK_CONDITION command) to SLP or HALT instruction and breaks.
   (c) When the user presses a BREAK key during SLP or HALT instruction execution.

(2) Table 9-1 shows the relationship between the execution mode of GO command and specifying the execution condition with BREAK or TRACE_BREAK_CONDTION command. Be sure that there are some combinations not to be specified.

Table 9-1 Relationship between GO command and BREAK or TRACE_BREAK_CONDITION command

No.	Execution mode of GO command	TRACE_BREAK_CONDITION				Breakpoint specification
		Starting condition	Range	Break condition1	Break condition2	
1	Normal mode (no option)	o	o	o	o	o
2	Trigger mode (option T, NT)	x	x	o	o	x
3	High-speed trace mode (option H, NH)	x	x	o	o	x
4	Cycle reset mode (option R, NR)	x	x	x	x	x

o --- Specified condition is valid.
x --- Specified condition is invalid. The same as the case without any condition specified.

(3) As for the high-speed trace mode, trace information save timing is synchronous to CLOCK signal of the ASE, not to CLOCK signal of the user system.

(4) When the stack pointer is not in RAM area if the operation stops at the breakpoint, the ASE cannot detect that the operation stops at the breakpoint. Therefore, the ASE stops regarding that RST $30 instruction in the user program is executed.

(5) During SLP or HALT instruction execution, pressing a BREAK key cancels SLEEP or HALT state with NMI. Therefore, PC becomes $0066.

(6) When a break condition is satisfied during DMA transfer (burst mode), the ASE doesn't break immediately, but after DMA transfer the ASE executes the first instruction and breaks.

(7) When a break condition is satisfied during DMA transfer (cycle steal mode) DMA stops.(The ASE inputs NMI when stopping the user program.)
When executes GO command again, the ASE makes DME bit ENABLE just before executing the user program. Therefore, when executing the user program, DMA of several bytes is transferred (not as trace information).

(8) When the condition of TRACE_BREAK_CONDITION command is specified with a logical address, the logical address checks if the condition is satisfied.

(9) When the user program execution stops at a breakpoint and restarts from the breakpoint, the ASE executes the instruction of the breakpoint by single-step and then continues real-time emulation. Trace information by single-step execution is not saved.
If the restart address is not just the breakpoint, the ASE restarts the user program in real time. (without single-step process)

(10) When the ASE executes the breakpoint with specification of continuing, GO command is continued, but a trace buffer is cleared at the breakpoint.

(11) If $\overline{\text{NMI}}$ signal is input just before GO command execution, NMI interrupt is accepted just after user program has started, and then the ASE executes NMI servicing (starting address $66). The MPU detects NMI by noticing the change from "High" to "Low", but note that the ASE always accepts NMI when the level is "Low" in inputting GO command. Also, when the ASE breaks in NMI servicing routine and NMI signal is still "Low" in restarting, the ASE restarts from $\overline{\text{NMI}}$ servicing routine again. So, please set $\overline{\text{NMI}}$ signal "High" before restarting.

Example

(1) When executes the user program from $1000;

```
:GO 1000 (CR)
 PROCESS STARTED FROM 1000 01000
```

(2) When executes the user program from $1000 in the high-speed mode and stops with BREAK CONDITION 1 (set with TRACE_BREAK_CONDITION command);

```
:GO 1000;H (CR)
PROCESS STARTED FROM 1000 01000
PC=100C 0100C SP=EFFF 0EFFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=CA  F=80=S.*.*... B=00  C=00  D=00  E=00  H=12  L=00 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=51
RUN-TIME=&0000H:00M:00S:002090US
BREAK CONDITION 1
```

(3) When executes the user program from $1000 in the trigger mode and stops with a BREAK key pressed (Interactive input format);

```
:GO ? (CR)
START ADDRESS (RESET/addr/return) ?1000  (CR)
MODE HIGH SPEED(H),TRIGGER(T),CYCLE RESET(R),FROM NMI(N)
(H/T/R/N/NH/NT/NR/return) ?T (CR)
PROCESS STARTED FROM 1000 01000
PC=1008 01008 SP=EFFF 0EFFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=C9  F=80=S.*.*... B=4C  C=00  D=00  E=00  H=11  L=B4 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=35
RUN-TIME=&0000H:11M:04S:915170US
BREAK KEY
```

(4) When continuously executes from the address pointed by the program counter during NMI processing;

```
:GO ;N (CR)
 PROCESS STARTED FROM 1000 01000
```

(5) When specifies the RESET processing to the user program and executes the cycle reset every 1mS (Interactive input format);

```
:GO ? (CR)
START ADDRESS (RESET/addr/return) ?RESET (CR)
MODE HIGH SPEED(H),TRIGGER(T),CYCLE RESET(R),FROM NMI(N)
(H/T/R/N/NH/NT/NR/return) ?R (CR)
CYCLE TIME 10U,100U,1M,10M,100M,655MS (1-6) ?3 (CR)
PROCESS STARTED FROM 0000 00000
*** RESET IN BY ASE !
```

9.20 HELP — DISPLAY OF ALL COMMANDS

Command Syntax

HELP (CR)

Function

(1) Displays all commands and their abbreviation.

Description

(1) The following contents are shown.

```
:HELP (CR)
.PC      .SP      .A       .B       .C       .D       .E       .H       .L       .F
.IX      .IY      .ZA      .ZB      .ZC      .ZD      .ZE      .ZH      .ZL      .ZF

AD: ADDRESS          AM: ADDRESS_MODE     A  : ASSEMBLE          BU: BACKUP
BS: BASE             B  : BREAK           BM: BREAK_MASK         CH: CHECK
CL: CLOCK            CC: COMMAND_CHAIN    CV: CONVERT            C  : COVERAGE
DA: DISASSEMBLE      DP: DISPLAY          FF: FLOPPY_FORMAT      FL: FLOPPY_LOAD
FS: FLOPPY_SAVE      FV: FLOPPY_VERIFY    G  : GO                HE: HELP
H  : HOST            I  : IO_CONTROL      L  : LOAD              MP: MAP
M  : MEMORY          MV: MOVE             MR: MOVE_TO_RAM        P  : PRINT
Q  : QUIT            R  : REGISTER        RT: RESULT             SV: SAVE
ST: STATUS           S  : STEP            SP: STOP               SY: SYMBOL
T  : TRACE           TC: TRACE_BREAK_CONDITION                   TP: TRACE_POINTER
V  : VERIFY
```

(2) Interactive input format

Display is the same as that of (1).

9.21 HOST — CONNECTION TO A HOST SYSTEM

Command Syntax

HOST[△{<drive name> : ASE180}] CR

Function

(1) SD200 CP/M-68K system loads and activates the program for ASE support.
(2) I/O control data to a host system is specified, so that the user can connects the ASE to a host system except SD200.
(3) Program for ASE support of SD200 is terminated.

Description

(1) When the ASE interface program is loaded to SD200 CP/M-68K.

```
:HOST B:ASE180 CR
```

In this case, ASE interface program (file name is ASE180.68K) is regarded as in drive B.

By this command, the ASE interface program in a floppy disk is loaded and it starts. If it starts, the following message is displayed.

```
HD64180  ASE INTERFACE (S180CAS1F)  Vn.m*
 COPYRIGHTED BY HITACHI, LTD.  1985.
```

*: version number

(2) When a host system except for SD200 CP/M-68K is connected, response requirement messages to decide the baud rate, data type and control method to the host system are displayed. Input some key to meet the requirements of the host system.
Display contents are as follows.

```
:HOST CR
<former condition>
BAUD RATE
1:19200  2:9600   3:4800  4:2400  5:1200  6:600  7:300
KEY IN (1 to 7)  ? ① CR
CHARACTER LENGTH  (7/8)  ? ② CR
PARITY (E/O/D)  ? ③ CR
STOP BIT (1/2)  ? ④ CR
X-ON, X-OFF CONTROL (Y/N)  ? ⑤ CR
```

<former condition> is displayed as x x x x x. Each value is same as ① to ⑤ above. (①②③④⑤)

① --- The number of the baud rate is specified.(initial value "2")
② --- The data bit number of one character is specified.(initial value"8")
③ --- Parity is specified.(initial value "D")
E : even parity
O : odd parity
D : no parity
④ --- Stop bit (initial value "1")
1 : 1 stop bit
2 : 2 stop bits
⑤ --- Whether control X-ON and X-OFF or not is specified.
(initial value "Y")
Y : control of X-ON and X-OFF is accomplished.
N : control of X-ON and X-OFF is not accomplished.

(4) By inputting this command, commands of the following table 9-2 can be used.

Table 9-2 Host system usage command

No.	Command name	Function	Host
1	LOAD	load of a user program	SD200, the others
2	SAVE	save of memory contents	SD200, the others
3	VERIFY	verify of a user program	SD200, the others
4	COMMAND_CHAIN	input from SD200 of ASE command	SD200
5	PRINT	starting the printer connected to SD200	SD200

(5) When terminating the ASE interface program on SD200

```
:HOST - CR
```

(6) Interactive input format

(a) Opening a file with SD200 host system

```
:HOST ? CR
OPEN OR CLOSE (O/C) ? O CR
HOST IS SD200 (Y/N) ? Y CR
FILE NAME ? ASE180 CR
```

(b) Opening a file with a host system other than SD200

```
:HOST ? CR
OPEN OR CLOSE (O/C) ? O CR
HOST IS SD200 (Y/N) ? N CR
<former condition>
```

(c) Closing the opened file

```
:HOST ? CR
OPEN OR CLOSE (O/C) ? C CR
```

Note

(1) If a file I/O error occurs on SD200, an error message of the host system is output. About the contents of error messages, refer to "CP/M-68K operating system user's guide".

(2) Do not input an unnecessary key after command input. If do so, the host system might not get started-up.

Example

(1) When activates the ASE interface program in the drive B onto the SD200 CP/M-68K system;

```
:HOST B:ASE180 (CR)
```

(2) When terminates the ASE interface program on the SD200;

```
:HOST - (CR)
```

(3) When connects the host system except SD200 CP/M-68K to the ASE (Interactive input format);

```
:HOST ? (CR)
 OPEN OR CLOSE (O/C) ?O (CR)
 HOST IS SD200 (Y/N) ?N (CR)
 28D1Y
 BAUD RATE
 1:19200 2:9600 3:4800 4:2400 5:1200 6:600 7:300
 KEY IN (1-7) ?3 (CR)
 CHARACTER LENGTH (7/8) ?8 (CR)
 PARITY (E/O/D) ?D (CR)
 STOP BIT (1/2) ?2 (CR)
 X-ON, X-OFF CONTROL (Y/N) ?Y (CR)
```

9.22 IO_CONTROL — I/O CONTROL OF DATA

Command Syntax

IO_CONTROL [△INT = {E / D}] [△NMI = {E / D}] [△REF = {E / D}] (CR)

E : Input inhibit is cancelled
D : Input is inhibited

Function

(1) Input of $\overline{INT0}$ to $\overline{INT2}$ signals into MPU is inhibited.
(2) Input of $\overline{NMI}$ signal into MPU is inhibited.
(3) At the execution time, saving of refresh cycle trace data is inhibited.
(4) Specified conditions are displayed.

Description

(1) The details of each condition are as follows.
(a) Control of $\overline{INT0}$ to $\overline{INT2}$ signals

INT = {E / D}
E : Input inhibit of $\overline{INT0}$ to $\overline{INTO2}$ signals is cancelled
D : Input of $\overline{INT0}$ to $\overline{INT2}$ signals is inhibited

When $\overline{INT0}$ to $\overline{INT2}$ signals are "Low", the signal is not input into MPU by "D" specification. That is, interrupt by $\overline{INT0}$ to $\overline{INT2}$ signals does not occurred.

(b) Control of $\overline{NMI}$ signal

NMI = {E / D}
E = Input inhibit of $\overline{NMI}$ signal is cancelled.
D = Input of $\overline{NMI}$ signal is inhibited

If $\overline{NMI}$ signal is "Low", the signal is not input into MPU by "D" specification. That is, interrupt by $\overline{NMI}$ signal does not occurred.

(c) Save control of trace data of a refresh cycle

REF = {E / D}
E : Trace data of a refresh cycle is saved.
D : Trace data of a refresh cycle is not saved.

When $\overline{REF}$ signal is a low level, bus data are not saved into a trace buffer by "D" specification. Save control is used in the case in which the data of a refresh cycle are not required like a software debugging.

(2) Specification order of each condition is not fixed.

(3) As for unspecified condition, do not change the current one.
So, specify as follows.

```
:IO_CONTROL   INT = D  NMI = E   REF = E  CR
```

Then, specify as follows.

```
:IO_CONTROL   NMI = D  CR
```

The above specification is the same as the following specification.

```
:IO_CONTROL   INT = D   NMI = D   REF = E  CR
```

(4) Specified condition is displayed as follows.

```
:IO_CONTROL CR

 INT = D   NMI = E   REF = E
```

(5) Interactive Input format

(a) Setting

```
:IO_CONTROL  ? CR
SET OR DISPLAY (S/D) ?  S CR
INT0-2 ENABLE OR DISABLE (E/D/return)  ?  [{E/D}] CR
NMI    ENABLE OR DISABLE (E/D/return)  ?  [{E/D}] CR
REFRESH CYCLE TRACE ENABLE OR DISABLE (E/D/return) ?  [{E/D}] CR
```

(b) Display

```
:IO_CONTROL    ? CR
SET OR DISPLAY  (S/D) ?   D   CR
INT = (1)  NMI = (1)  REF = (1)
 (1) displays "E" or "D"
```

Note

(1) With the ASE started-up, the default is ENABLE state.

(2) In the interactive input format, if omits all items (only return input), the \SE displays the setting contents and goes to command input wait state.

(3) Even if specifies "REF = D", during user program execution in high-speed trace mode, refresh cycle is saved in the trace buffer.

Example

(1) When inhibits the input of $\overline{INT0}$ to $\overline{INT2}$ signal and displays the current state;

```
:IO_CONTROL INT=D CR
:IO_CONTROL CR
 INT=D NMI=E REF=E
```

(2) When prevents from getting the trace information of refresh cycle;

```
:IO_CONTROL REF=D CR
```

(3) When inhibits the input of $\overline{NMI}$ signal in interactive input format

```
:IO_CONTROL ? CR
 SET OR DISPLAY (S/D) ?S CR
 INT0-2 ENABLE OR DISABLE (E/D/return) ?E CR
 NMI    ENABLE OR DISABLE (E/D/return) ?D CR
 REFRESH CYCLE TRACE ENABLE OR DISABLE (E/D/return) ?E CR
```

(4) When displays the current state in interactive input format;

```
:IO_CONTROL ? CR
 SET OR DISPLAY (S/D) ?D CR
 INT=E NMI=D REF=E
```

9.23 L O A D — LOAD THE OBJECT PROGRAM

Command Syntax

```
LOAD{ △<file name>                    }
    { △[<value>][;{S} ]]}          } CR
                 {H}
```

<VALUE> ; Additional value to an address
S ; S type format
H ; Intellex HEX format (default)

Function

(1) Loads the user program into the user memory from the host system.

Description

(1) In the case of connecting the SD200 CP/M-68K host system.
CP/M-68K 64180 cross macro assembler makes an object program composed of some files as shown below. Refer to "CP/M-68K 64180 Cross Macro Assembler User's Manual" for details.

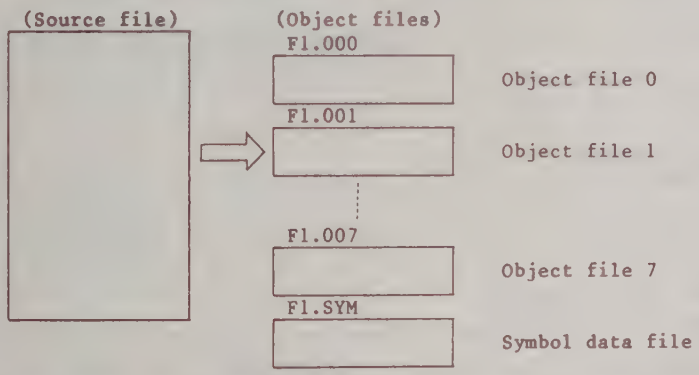


Fig. 9-10 Object File Output by CP/M-68K 64180 Cross Macro Assembler

(a) When loading all files, abbreviate the subfile names. The ASE changes the subfile names to 000 - 007, SYM and loads them in order.
(Example)
:LOAD B:F1 CR
Loads from the files F1.000, F1.001, --- F1.007, F1.SYM.
The ASE loads the symbol name, but don't load the symbol name using characters except alphanumerics.

(b) When loading a particular file, input subfile name.
(Example)
:LOAD A:F1.006 CR
Loads from the file F1.006.

(2) In the case of connecting a general purpose host system.
The format of the object program the ASE loads is the program in address 0 to $FFFF. Since the HD64180 has a memory space of 512KB, you have to input the value to be added to the address data in the object program. The object program is loaded in these addresses.

:LOAD [<value>][;{S / H}] CR

<value> ---- Value to be added to the address data in the object program. If nothing is typed, 0 is input.
S ---- S-type format
H ---- INTELLEC HEX format (default)

Transmit data from the host system after inputting this command.

(3) When the load has been finished, the starting address and end address are displayed.

```
TOP ADDRESS    xxxx xxxxx
END ADDRESS    xxxx xxxxx
                  ①    ②
```

① Logical address
② Physical address

(4) Interactive input format
(a) SD200 CP/M-68K host system

```
:LOAD ?CR
 FILE NAME ? <file name>CR
```

(b) General purpose host system

```
:LOAD ?CR
 ADDITIONAL ADDRESS (addr/return) ? [<value>]CR
 FORMAT IS S-TYPE OR HEX - TYPE (S/H) ? {S / H} CR
```

Notes

(1) After loading, memory contents and file contents are not verified. Use VERIFY command to verify.
(2) <TOP ADDRESS> displays the low-order address which has been loaded, and <END ADDRESS> does the high-order address.
(3) Refer to "4.3 Connecting with General Purpose Host System" for details.
(4) Take notice of the following, except when the host system is SD200.
(a) If an error occurs during data transfer, the ASE displays an error message and goes into the command input wait state. At this time, the ASE doesn't accept data from the host system, but in the case of the check sum error, it displays the next message and operates according to your answer.

```
CHECK SUM ERROR  ADDR = xxxx
                        ②
CONTINUE  (Y/N)? ①CR
```

① --- Y : Continues the operation disregarding the error.
N : Stops the operation and goes into the input wait state of the next command.
② --- The value of the location address in a record where an error occur.

(b) If the host system is not connected or the host system transmit no data, the ASE doesn't respond. In this case, pressing a BREAK key causes the ASE to stop processing and the ASE goes to command input wait state.

Example

(1) When loads all object files having "OBJF" as a file name in the disk drive B in connecting to SD200 CP/M-68K host system (note that the object file is prepared by CP/M-68K cross macro assembler);

```
:LOAD B:OBJF (CR)
TOP ADDRESS : 0000 0000U
END ADDRESS : E234 7F234
```

(2) When loads only the specified file in connecting to SD200 CP/M-68K host system;

```
:LOAD ? (CR)
FILE NAME ?B:SAMPLE.LO (CR)
TOP ADDRESS : 1000 01000
END ADDRESS : 17FF 017FF
```

(3) When loads the object program from $0 to $7FF in INTELLEC HEX format in connecting to a general purpose host system;

```
:LOAD (CR)
TOP ADDRESS : 0000 00000
END ADDRESS : 07FF 007FF
```

(4) When loads the object program from $800 to $FFF added -$100 offset in S-type format in connecting to a general purpose host system (Interactive input format);

```
:LOAD ? (CR)
ADDITIONAL ADDRESS (addr/return ) ?7FF00 (CR)
FORMAT IS S-TYPE OR HEX-TYPE (S/H) ?S (CR)
TOP ADDRESS : 0700 00700
END ADDRESS : 0EFF 00EFF
```

9.24 MAP SELECT OR DISPLAY MEMORY ATTRIBUTES

Command Syntax

```
MAP [△<address 1>△<address 2>[= { U  } ]] CR
                                { S  }
                                { UR }
                                { SR }
```

address 1 ; starting address (multiple of $1000 or 0)
address 2 ; end address (multiple of $1000 -1)

U	; user memory
S	; emulation memory
UR	; write protect user memory
SR	; write protect emulation memory

Function

(1) Selects the attributes of memory area
(2) Displays the selected attributes

Description

(1) Selecting the attributes of the memory area
: MAP <address 1> <address 2> ; <attributes>CR

<address 1> ; starting address of the memory
<address 2> ; end address of the memory
<attributes>; attributes to be selected

U	; user memory
S	; emulation memory
UR	; write protect user memory
SR	; write protect emulation memory

When selecting S (emulation memory), the remaining emulation memory capacity is displayed.

```
REMAINS SYSTEM MEMORY  &xxxKB
                       (1)
```

(1) remaining memory capacity

(2) Displaying the attributes of the memory area

```
: MAP <address 1> <address 2> CR
  xxxx xxxxx-xxxx xxxxx=xx xxxx xxxxx-xxxx xxxxx=xx  ---------
       ①          ②
```

① address area (Both logical and physical address are displayed.)
② attributes (The same abbreviations as those at setting are displyed.)

(3) Displaying the memory attributes of all memory areas

```
: MAP CR
  xxxx xxxxx-xxxx xxxxx=xx xxxx xxxxx-xxxx xxxxx=xx -----
    ¦   ①          ¦    ②  ¦               ¦
    ¦              ¦       ¦               ¦
    ¦              ¦       ¦               ¦
  REMAINS SYSTEM MEMORY   &xxxKB
                          ③
```

① address area (Both logical and physical address are displayed.)
② attributes (the same abbreviations as those at setting are displayed.)
③ remaining emulation memory capacity

(4) Interactive input format

(a) Setting

```
: MAP ? CR
  SET OR DISPLAY (S/D)?  S CR
  TOP ADDRESS ? <address> CR
  END ADDRESS ? <address> CR
  ATTRIBUTE USER(U), ASE SYSTEM(S), READ ONLY(R) (U/S/UR/SR) ?  x CR
  [<remaining emulation memory capacity>]
```

(b) Partial display

```
: MAP ? CR
  SET OR DISPLAY (S/D)? D CR
  ALL       AREA (Y/N) N CR
  TOP ADDRESS ? <address> CR
  END ADDRESS ? <address> CR
```

(c) Displaying all memory areas

```
: MAP ? CR
  SET OR DISPLAY (S/D) ? D CR
  ALL AREA (Y/N)? Y CR
```

Note

(1) When selecting a memory area, specify it by every 4KB.
(2) If the starting address is not a multiple of $1000, cut it down; if the end address is not (the multiple of $1000)-1, raise it.
At this time, the following message is displayed on the screen.

```
*** 52:MAPPING BOUND MUST 4KB
```

Note that the above specification is already done.
(3) In the initial state, all memory area are memory attributes of the user system.
(4) When specifying memory attribute by a logical address, physical address in the area should be continuous. (Refer to section 6.2.4.)

Example

(1) When the emulation memory is from $0 to $FFFF;

```
:MAP 0 FFFF=S (CR)
 REMAINS SYSTEM MEMORY &456KB
```

(2) When displays all the current memory attribute;

```
:MAP (CR)
 0000 00000-FFFF 0FFFF=S  **** 10000-**** 7FFFF=U
 REMAINS SYSTEM MEMORY &456KB
```

(3) When write protect emulation memory is from $0 to $7FFF in interactive input format;

```
:MAP ? (CR)
 SET OR DISPLAY (S/D) ?S (CR)
 TOP ADDRESS ?0    (CR)
 END ADDRESS ?7FFF (CR)
 ATTRIBUTE USER(U),ASE SYSTEM(S),READ ONLY(R) (U/S/UR/SR) ?SR
 REMAINS SYSTEM MEMORY &456KB
```

(4) When displays all memory attribute in interactive input format;

```
:MAP ? (CR)
 SET OR DISPLAY (S/D) ?D  (CR)
 ALL AREA (Y/N) ?Y  (CR)
 0000 00000-7FFF 07FFF=SR 8000 08000-FFFF 0FFFF=S  **** 10000-**** 7FFFF=U
 REMAINS SYSTEM MEMORY &456KB
```

9.25 MEMORY — DISPLAY OR CHANGE THE MEMORY CONTENTS

Command Syntax

```
                ( <address 1> Δ { <address 2>  [;I] }
MEMORY Δ        {               { <byte number>  }          CR
                ( <address 3> [;<option>]
```

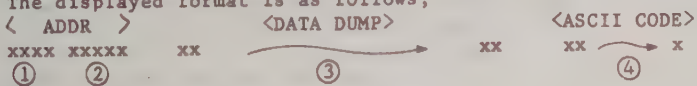
address 1 ; starting address to be displayed
address 2 ; end address to be displayed
byte number; byte number to be displayed
address 3 ; memory address to be changed
option ; W ... display and change in the unit of word
O ... display and change of odd address (in byte)
V ... display and change of even address (in byte)
N ... not verify
I ... display and change of I/O space(address 0 to address $FFFF)
not specified ... display and change in the unit of byte

Function

(1) Displays and changes the memory contents

Description

(1) Displaying the memory contents
(a) Displaying the memory contents of the specified bytes from the specified address. The starting address should be greater than the byte number.
: MEMORY <starting address> <byte number> [;I] CR
(b) Displaying the memory contents between the specified addresses. The starting address should be smaller than or equal to the end address.
: MEMORY <starting address> <end address> [;I] CR
(c) The displayed format is as follows;



① logical address
② physical address
③ memory contents displayed in hexadecimal
④ memory contents displayed by ASCII (if there is not a suitable character, a period represents it.)
(d) When option I is selected, the memory contents of I/O space is displayed.

(2) Changing the memory contents

:MEMORY <address> [;<option>] CR

(a) N can be selected with either one of W, O, V or I. In this case, be sure to select N first.

(Ex.)

:MEMORY 100;NW CR

(b) If this command is input, the contents of the specified address is displayed, and the system goes to the sub-command mode.

```
xxxx xxxxx xx   ? <sub-command> CR
           xxxx
①    ②    ③        ④
```

① logical address
② physical address
③ memory contents (in the case of W option, 2 bytes are displayed.)
④ sub-command

Sub-command format is as follows;

[<data>] [;<option>] CR

data ; Data to be changed should be selected.

option ; B ... display or change in the unit of byte
W ... display or change in the unit of word
O ... display and change of odd address (in the unit of byte)
V ... display and change of even address(in the unit of byte)
^ ... display the memory contents of the previous address
= ... display the current address contents again after change
. ... terminate MEMORY command

(i) [[<data>];] W CR ... [after change] display in the unit of word
(ii) [[<data>];] O CR ... [after change] display only odd address
(iii) [[<data>];] V CR ... [after change] display only even address
(iv) [[<data>];] ^ CR ... [after change] display the contents of the previous address
(v) [[<data>];] = CR ... [after change] display the current address contents again
(vi) [[<data>];] . CR ... [after change] terminate MEMORY command
(vii) [<data>]; B CR ... [after change] display in the unit of byte
(viii) [<data>] CR ... [after change] display the next address contents

(3) Interactive input format

(a) Displaying the memory contents

```
: MEMORY ? CR
  MEMORY CHANGE OR DISPLAY (C/D) ? D CR
  TOP ADDRESS ?  <address> CR
  END ADDRESS OR SIZE ?  <address or byte number > CR
  I/O AREA OR MEMORY (I/M)  ?  x  CR
    In the case of I, I/O space is displayed.
```

(b) Changing the memory contents

```
: MEMORY ? CR
  MEMORY  CHANGE OR DISPLAY (C/D) ? C CR
  ADDRESS ? <address> CR
  WORD(w), ODD(O), EVEN(V), I/O AREA(I), NOT VERIFY(N)
  (W/O/V/I/N/NW/NO/NV/NI/return)? [x] CR
  xxxx xxxxx <data> ?..... sub-command wait state
```

Note

(1) Even if the byte number is specified as 0, 1-byte data is displayed.

(2) W option should be selected with even address. If selected with odd address, the address is decreased by 1 and displayed.

(3) I option should be selected always with display in byte unit. If you would like to change the display format, after the ASE goes to sub command mode, change it.

(4) When the contents in the address area where physical addresses are not sequential are displayed in logical address input mode, at the address which is not sequential to the former address, the new line is fed then the contents are displayed.

Example

(1) When displays the memory contetns from $1000 to $103F;

```
:MEMORY 1000 103F (CR)
<  ADDR  >  <                    DATA    DUMP                        >  <  ASCII CODE  >
1000 01000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
1010 01010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
1020 01020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
1030 01030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

(2) When displays the memory contents of $40 bytes form $1000;

```
:MEMORY 1000 40 (CR)
<  ADDR  >  <                    DATA    DUMP                        >  <  ASCII CODE  >
1000 01000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
1010 01010  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
1020 01020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
1030 01030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
```

(3) When displays I/O space form $0 to $3F (interactive input format);

```
:MEMORY ? (CR)
MEMORY CHANGE OR DISPLAY (C/D) ?D (CR)
TOP ADDRESS ?0 (CR)
END ADDRESS OR SIZE ?3F (CR)
I/O AREA OR MEMORY (I/M) ?I (CR)
<  ADDR  >  <                    DATA    DUMP                        >  <  ASCII CODE  >
0000 00000  10 00 27 87 04 02 FF FF  FF FF 0F FF FF FF FF FF  ..'.............
0010 00010  00 FF FF FF FF FF FF FF  23 FF FF FF FF FF FF FF  ........#.......
0020 00020  FF FF 0F FF FF 0F FF FF  FF FF 0F FF FF 00 FF FF  ................
0030 00030  32 C1 F0 00 39 FF FC FF  00 00 F0 FF FF FF FF 1F  2...9...........
```

(4) When changes the memory contents form $1000;

```
:MEMORY 1000 (CR)
1000 01000 00   ? 41 (CR)
1001 01001 00   ? 53 (CR)
1002 01002 00   ? 54 (CR)
1003 01003 00   ? ^ (CR)
1002 01002 54   ? 45 (CR)
1003 01003 00   ? . (CR)
```

(5) When changes the memory contents by every 2 bytes from $2000;

```
:MEMORY ? (CR)
MEMORY CHANGE OR DISPLAY (C/D) ?C (CR)
ADDRESS ?2000 (CR)
WORD(W),ODD(O),EVEN(V),I/O AREA(I),NOT VERIFY(N)
(W/O/V/I/N/NW/NO/NV/NI/return) ?W (CR)
2000 02000 0000 ? 7FFF (CR)
2002 02002 0000 ? 9000 (CR)
2004 02004 0000 ? ^ (CR)
2002 02002 9000 ? 8000;. (CR)
```

Example

(6) When changes the memory contents without verify after write from $2100;

```
:MEMORY 2100;N (CR)
 2100 02100 00    ? 80 (CR)
 2101 02101 00    ? 81 (CR)
 2102 02102 00    ? 82 (CR)
 2103 02103 00    ? 93;^ (CR)
 2102 02102 82    ? 92;^ (CR)
 2101 02101 81    ? 91;^ (CR)
 2100 02100 80    ? 90;. (CR)
```

(7) When changes the memory contents for odd addresses from $2201;

```
:MEMORY ? (CR)
 MEMORY CHANGE OR DISPLAY (C/D) ?C (CR)
 ADDRESS ?2201 (CR)
 WORD(W),ODD(O),EVEN(V),I/O AREA(I),NOT VERIFY(N)
 (W/O/V/I/N/NW/NO/NV/NI/return) ?O (CR)
 2201 02201 00    ? 1 (CR)
 2203 02203 00    ? 2 (CR)
 2205 02205 00    ? 3 (CR)
 2207 02207 00    ? . (CR)
```

9.26 MOVE — TRANSFER THE MEMORY CONTENTS

Command Syntax

```
MOVE  △<address 1> △<address 2> △<address 3>  CR
```

address 1 ; starting address to be transferred
address 2 ; end address to be transferred
address 3 ; destination address

Function

(1) Transfers the memory contents in specified area to the specified address

Description

(1) Transfers the memory to the specified address. If verify error occurs in transferring the memory, outputs the following message, then terminates this command.

```
FAILED AT <address>  WRITE=<write address>  READ=<read data>
```

(2) Interactive input format

```
: MOVE ? CR
  TOP ADDRESS ? <address> CR
  END ADDRESS ? <address> CR
  DESTINATION ADDRESS ? <address> CR
```

Note

(1) The data in I/O space cannot be transferred.
(2) The destination area should be write enable area .
(3) The address range to be transferred should have sequential physical address range. (Refer to section 6.2.4.)
Note that in the destination area there should be sequential physical address area from the destination address.

Example

(1) When transfers the memory area from $1000 to $10FF into that from $2000;

```
:MOVE 1000 10FF 2000 CR
```

(2) When transfers the memory area from $2100 to $21FF into that from $1100 in interactive input format;

```
:MOVE ?  CR
 TOP ADDRESS ?2100 CR
 END ADDRESS ?21FF CR
 DESTINATION ADDRESS ?1100 CR
```

9.27 MOVE_TO_RAM — MOVE ROM CONTENTS TO THE ASE EMULATION MEMORY

Command Syntax

MOVE_TO_RAM △ <address 1> △ <address 2> CR

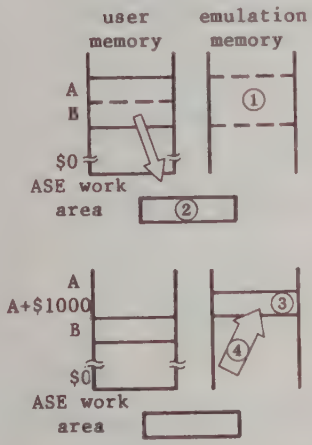
Function

(1) Moves the user memory contents in the range of specified addresses to the emulation memory. After moving the contents, the specified area has the emulation memory attribute; it is writable into the emulation memory.

Description

(1) This command is used for changing and executing the ROM contents in the user system. The processing flow is as follows;

:MOVE_TO_RAM A B CR



① Check whether the emulation memory has space for the ROM to be moved or not.

② Read 4KB of ROM into the ASE working area.

③ Change the attribute of the read ROM address area to the emulation memory area.

④ Move the contents of the ASE working area onto the emulation memory area.

⑤ Repeat ② to ④ to move all contents of the ROM onto the emulation memory.

Fig. 9-11 MOVE_TO_RAM Command Operation Procedure

: MOVE_TO_RAM <address 1> <address 2> CR
REMAINS SYSTEM MEMORY <remaining emulation memory capacity>

address 1 : top address to be transferred (0 or a multiple of $1000)

address 2 : end address to be transferred (a multiple of $1000 -1)

remaining emulation memory capacity : The remaining emulation memory capacity is displayed in decimal.

(2) Interactive input format

: MOVE_TO_RAM ? CR
TOP ADDRESS ? <address 1> CR
END ADDRESS ? <address 2> CR
REMAINS SYSTEM MEMORY <remaining emulation memory capacity>

Notes

(1) When specifying a memory area, the minimum unit should be 4KB.
If the starting address is not a multiple of $1000, cut it down.
If the end address is not the multiple of $1000 -1, raise it.

(2) The original attribute of the specified area does not matter. (It is not necessary to be the user memory.)

(3) Before inputting this command, be sure that the emulation memory has space enough for the specified area to be moved.

(4) When specifying the address by a logical address, physical address in the area should be sequential. (Refer to section 6.2.4.)

(5) This command does not verify. If necessary, the applied area should be saved by FLOPPY_SAVE command first. Then, after transferring the memory, verify with FLOPPY_VERIFY command.

Example

(1) When moves the contents of user memory from $0 to $1FFF to emulation memory;

```
:MOVE_TO_RAM 0 2000-1 (CR)
REMAINS SYSTEM MEMORY &512KB
:MAP (CR)
0000 00000-1FFF 01FFF=S  2000 02000-**** 7FFFF=U
REMAINS SYSTEM MEMORY &512KB
```

(2) When moves the contents of user memory from $2000 to $3FFF to emulation memory in interactive input format;

```
:MOVE_TO_RAM ? (CR)
TOP ADDRESS ?2000 (CR)
END ADDRESS ?4000-1 (CR)
REMAINS SYSTEM MEMORY &512KB
:MAP (CR)
0000 00000-1FFF 01FFF=U  2000 02000-3FFF 03FFF=S  4000 04000-**** 7FFFF=U
REMAINS SYSTEM MEMORY &512KB
```

9.28 PRINT — PRINTER CONNECTION

PRINT [△{H / L / -}] CR

H : Assigning the SD200 printer
L : Assigning the ASE printer
- : Disassigning the printer

Function

(1) Assign a printer of the ASE station or SD200 for outputting the ASE command result.
(2) Disassign the printer.

Description

(1) Assigning the ASE printer connected to the ASE.
:PRINT L CR
(2) Assigning the SD200 printer
:PRINT H CR
(3) Disassigning the printer
:PRINT - CR
(4) Interactive input format
(a) Assigning the printer
:PRINT ? CR
ASSIGN OR CLOSE (A/C) ? A CR
LOCAL PRINTER OR HOST (L/H) ?{L / H} CR
(b) Disassigning the printer
:PRINT ? CR
ASSIGN OR CLOSE (A/C) ? C CR

Notes

(1) Execution result of the ASE command is also displayed on the console after assigning the printer. Note that even in the screen display scroll or screen display page mode with SD200 connected, the result is printed on the printer in the normal display scroll mode.
(2) An error message output from the host system is output only to the console but not to the printer.
(3) The printer of the SD200 and the ASE station are not assigned at the same time.

Example

(1) When starts the printer of the ASE station;

```
:PRINT L CR
```

(2) When starts the printer connected with SD200;

```
:PRINT H CR
```

(3) When disconnects the printer;

```
:PRINT - CR
```

(4) When starts the printer of the ASE station in interactive input format;

```
:PRINT ? CR
 ASSIGN OR CLOSE (A/C) ?A CR
 LOCAL PRINTER OR HOST (L/H) ?L CR
```

9.29 QUIT — TERMINATE THE ASE

Command Syntax

```
QUIT  CR
```

Description

(1) Terminating the ASE command processing mode and outputting the following message;

```
START ASE
  S : START ASE
  R : RELOAD & START ASE
  H : HOST TERMINAL MODE
      (S/R/H)  ?
```

Refer to chapter 7 for processing after this command.

(2) Interactive input format

```
: QUIT ? CR
  ASE END (Y/N) ? x CR
```

Example

(1) When terminates the ASE command processing mode;

```
:QUIT CR
 *** RESET IN BY ASE !

 START ASE
  S : START ASE
  R : RELOAD & START ASE
  H : HOST TERMINAL MODE
      (S/R/H) ?H CR
```

9.30 REGISTER — DISPLAY ALL REGISTERS

Command Syntax

REGISTER (CR)

Function

(1) Displays the contents of all registers

Description

(1) The contents of all registers are displayed in the following format.

```
PC=xxxx xxxxx  SP=xxxx xxxxx  IX=xxxx  IY=xxxx  IR=xx  C1LA=xxxx BKLA=xxxx  IF=x
 A=xx  F=xx=SZ*H*VNC   B=xx    C=xx  D=xx   E=xx   H=xx   L=xx CMBR=xxooo I=xx
ZA=xx ZF=xx=SZ*H*VNC  ZB=xx   ZC=xx ZD=xx  ZE=xx  ZH=xx  ZL=xx BKBR=xxooo R=xx
```

(2) Interactive input format

: REGISTER ? (CR)
<register display>

Example

(1) When displays the contents of all registers;

```
:REGISTER (CR)
PC=1009 01009 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=54=.Z*H*V..   B=00   C=00  D=28  E=00  H=20  L=00 CMBR=00000 I=00
ZA=00 ZF=00=...*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=62
```

9.31 RESULT — DISPLAY THE EXECUTION RESULT

Command Syntax

```
RESULT (CR)
```

Function

(1) Display the reason why the last GO or STEP command stopped, execution time and the current contents of each register.

Description

(1) The contents of all registers are displayed in the following format.

```
①  PC=xxxx xxxxx  SP=xxxx xxxxx  IX=xxxx  IY=xxxx  IR=xx  C1LA=xxxx BKLA=xxxx  IF=x
    A=xx  F=xx=SZ*H*VNC   B=xx    C=xx  D=xx   E=xx   H=xx   L=xx CMBR=xxooo I=xx
   ZA=xx ZF=xx=SZ*H*VNC  ZB=xx   ZC=xx ZD=xx  ZE=xx  ZH=xx  ZL=xx BKBR=xxooo R=xx

②  RUN-TIME=&xxxxH : xxM : xxS : xxxxxxUS
③  <Reason for command stop>
```

① ... The current contents of each register
② ... User program execution time. Display is in decimal numbers and up to about one year can be measured.
③ ... The reason why the last GO or STEP command stopped. See below;

Display	Meaning
(a) WRITE PROTECT	User program wrote into the write protect area.
(b) BREAK CONDITION 1	Command execution stopped at break condition 1.
(c) BREAK CONDITION 2	Command execution stopped at break condition 2.
(d) BREAK POINT	Command execution stopped at the breakpoint set by BREAK command.
(e) TRACE BUF FULL	Trace start condition is specified and command execution stopped after 2048 cycles of trace data is saved.
(f) STOP CONDITION	Command execution stops at the stop condition.
(g) BREAK KEY	Command execution stops by the BREAK key.
(h) RST $30 INST	RST $30 instruction executed.
(i) NO-EXECUTION	User program is not yet executed.
(j) STEP NORMAL END	Specified steps are executed by STEP command.
(k) SLP OR HALT INST	SLP or HALT instruction was executed by STEP command, so MPU was in wait state.
(l) PARITY ERROR	A parity error occurs with emulation memory access.

(2) Interactive input format

```
:RESULT (CR)
```

Display is the same as that of (1).

Note

(1) The ASE measures the execution time only with GO command. The execution time with STEP command will be 0.

Example

(1) When displays the result of execution;

```
:RESULT(CR)
PC=1009 01009 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=54=.Z*H*V..  B=00  C=00  D=28  E=00  H=20  L=00 CMBR=00000 I=00
ZA=00 ZF=00=...*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=62
RUN-TIME=&0000H:00M:00S:000018US
BREAK CONDITION 1
```

9.32 SAVE — OUTPUT MEMORY CONTENTS TO FILE

Command Syntax

SAVE [△<file name>] △<address 1> △<address 2> [;{S/H}] △[N]] ⓒR

file name : File name should be specified in the case of SD200.
address 1 : Starting address of the save area
address 2 : End address of the save area
S : S-type format
H : INTELLEC HEX format (default)
N : LF($0A) and four NULL($00) codes will be output after each record.

Function

(1) Transfer the memory contents up to 64KB to the host system.
(2) In the case of general host system, use S-type or INTELLEC HEX format (refer to Appendix A).
(3) In the case of SD200 CP/M-68K host system, data is transferred to a specified file.

Description

(1) In the case of the SD200 CP/M-68K host system, make a new file of a specified name in S-type format. In this case, the object format cannot be specified.

:SAVE <file name> <address 1> <address 2> ⓒR

The memory contents between address 1 and address 2 is output onto the <file name> on the SD200.

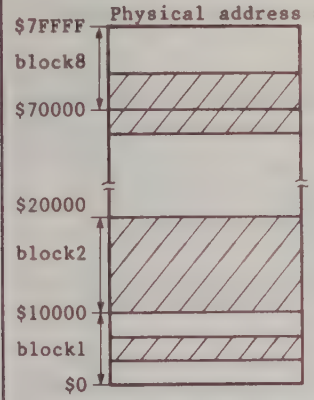
(2) In the case of the general host system, memory contents are output in a specified object format.

:SAVE <address 1> <address 2> [;[{S/H}] △[N]] ⓒR

S : S type format
H : INTELLEC HEX format (default)
N : LF ($0A) and four NULL ($00) codes will be output after each record.

The memory contents between address 1 and address 2 is output in a specified object format.

(3) The memory area to be output should be included in a block of 64KB. If it is more than 64KB, it should be divided to output.



Command example

:SAVE $6E000 $72000 CR

This will be an error because two blocks occupied.

:SAVE B:CPMOBJ.OBJ $10000 $1FFFF CR

The memory contents between $10000 and $1FFFF is output onto a file named CPMOBJ.OBJ (in the case of the SD200 CP/M-68K host system)

:SAVE $100 $5FF;S CR

The memory contents between $100 and $5FF is output in S-type format. (In the case of general purpose host system.)

Fig.9-12 Save of the memory contents

When outputting the memory area, location adress in the object module is output only two low-order bytes.

(4) Interactive input format

(a) Connecting with the SD200 CP/M-68K

```
:SAVE  ? CR
FILE NAME ?  <file name> CR
TOP ADDRESS ? <address>CR
END ADDRESS ? <address>CR
```

(b) Connecting with the general purpose host system

```
:SAVE  ? CR
TOP ADDRESS ? <address>CR
END ADDRESS ? <address>CR

FORMAT IS S-TYPE OR HEX-TYPE (S/H) ? {S|H} CR

LF($0A), NULL($00) OUT (Y/N) ? {Y|N} CR
```

Notes

(1) When the host system is the SD200 CP/M-68K, the new file is prepared with SAVE command, and you cannot save it by the same file name. In this case, once transfer the control to the CP/M-68K with QUIT command, then delete the file with ERA command.

(2) To specify the area to save in logical address mode, the physical address in the area should be sequential. (Refer to section 6.2.4.)

Example

(1) When makes a file with the contents of memory from $1000 to $17FF as a file name of "B:SAMPL1.LO" in connecting SD200 CP/M-68K host system;

```
:SAVE B:SAMPL1.LO 1000 17FF (CR)
```

(2) When makes a file with the contents of memory from $1800 to $1FFF as a file name of "B:SAMPL2.LO" under same condition of (1) above in interactive input format;

```
:SAVE ? (CR)
FILE NAME ?B:SAMPL2.LO (CR)
TOP ADDRESS ?1800 (CR)
END ADDRESS ?1FFF (CR)
```

(3) When transfers the contents of memory from $2000 to $27FF in S-type format to the general purpose host system;

```
:SAVE 2000 27FF;S (CR)
```

(4) When transfers the contents of memroy from $2800 to $2FFF in S-type format into host system under the same condition of (3) above in interactive input format;

```
:SAVE ? (CR)
TOP ADDRESS ?2800 (CR)
END ADDRESS ?2FFF (CR)
FORMAT IS S-TYPE OR HEX-TYPE (S/H) ?S (CR)
LF($0A),NULL($00) OUT (Y/N) ?Y (CR)
```

9.33 STATUS — DISPLAY THE ASE OPERATING STATUS

Command Syntax

```
STATUS (CR)
```

Function

(1) Displays the ASE operation status.

Description

(1) The displayed contents is the same as that is displayed in the full screen display mode. The display format is shown as follows;

```
1st line  space      CLOCK=x  MODE=x   SP=x  BR=&xxx   SYM=&xxxx   M=&xxxKB
            (a)        (b)     (c)     (d)   (e)        (f)         (g)
2nd line  HOST=xxxxx  TR-BR-COND=xxxxxx COVERAGE=x BASE=xx  INT=x  NMI=x  REF=x
            (h)          (i)            (j)       (k)     (l)    (m)   (n)
```

For details of each display, refer to "8.4 Full Screen Display Mode". Note that (a) displays the user execution status but always displays blank for this command display.

(2) Interactive input format

```
:STATUS ? (CR)
 <state display>
```

Example

(1) When displays the ASE operation status;

```
:STATUS (CR)
                        CLOCK=6  MODE=P  SP=0  BR=&0    SYM=&2     M=&456KB
    HOST=SD200  TR-BR-COND=  B1    COVERAGE=0    BASE=03  INT=E NMI=D REF=E
```

9.34 STEP — SINGLE STEP EXECUTION

Command Syntax

```
STEP [△[<starting address>]△[N]][;[<step number to be executed>△][{J|N}]]   CR
  or
  CR
```

starting address :"RESET" or an address should be specified.
N : single-step execution in NMI servicing.
J : branch instruction display mode
N : no trace display mode (specify after ";")

Function

(1) Execute a user program one by one from the specified address. Execution continues till the following conditions are satisfied.
(a) Stop conditions are satisfied.
(b) Specified steps execution has finished.
(c) BREAK key is pressed.
(d) An error occurs.
(2) Display the register contents and instruction mnemonic after each instruction execution. In the case of branch instruction display mode (J), only the branch instruction is displayed. In the case of no trace display mode (N) nothing is displayed during step execution until condition (1) aobve is satisfied.
(3) If inputting CR after STEP, the next command will be executed
:STEP ; 1 CR

Description

(1) The following is displayed after one step execution.

① PC=xxxx xxxxx SP=xxxx xxxxx IX=xxxx IY=xxxx IR=xx C1LA=xxxx BKLA=xxxx IF=x
A=xx F=xx=SZ*H*VNC B=xx C=xx D=xx E=xx H=xx L=xx CMBR=xxooo I=xx
ZA=xx ZF=xx=SZ*H*VNC ZB=xx ZC=xx ZD=xx ZE=xx ZH=xx ZL=xx BKBR=xxooo R=xx

② <execution instruction>
①... each register contents after execution
②... mnemonic of executed instruction
"DC xx" for inappropriate instruction

(2) Starting address is specified as follows;

```
[{RESET  |
 {address}]
```

RESET : After RESET signal input to the MPU, program execution starts from the reset address ($0).
address : Program execution starts from the specified address.
no specification : Program execution starts from an address the current PC register shows.

(3) When selecting the step number to be executed
:STEP [<starting address>][N] ; <step number to be executed> [{J|N}] CR

Set the step number after the starting address.

(4) In the case of branch instruction display mode

:STEP [<starting address>][N];[<step number to be executed>] J ©R

Program execution continues with the branch command displayed by J option.

(5) The following messages are displayed after STEP command is terminated.

① PC=xxxx xxxxx SP=xxxx xxxxx IX=xxxx IY=xxxx IR=xx C1LA=xxxx BKLA=xxxx IF=x
A=xx F=xx=SZ*H*VNC B=xx C=xx D=xx E=xx H=xx L=xx CMBR=xxooo I=xx
ZA=xx ZF=xx=SZ*H*VNC ZB=xx ZC=xx ZD=xx ZE=xx ZH=xx ZL=xx BKBR=xxooo R=xx

② <reason for execution stop>

①... Each register contents

②... The reason why STEP command stopped. See below;

Display	Meaning
(a) STOP CONDITION	Command execution stops after the stop condition is matched.
(b) STEP NORMAL END	Selected steps are executed by STEP.
(c) BREAK KEY	Command execution stops by the BREAK key.
(d) SLP OR HALT INST	SLP or HALT instruction was executed by STEP command, so MPU was in wait state.
(e) WRITE PROTECT	User program wrote into the write protect area.
(f) PARITY ERROR	A parity error occurs with emulation memory access.

(6) If the following error occurs during user program execution, the ASE displays the execution state. The display position is the same as that of GO command. In full screen display mode, it's displayed in the 21st line (which displays execution state) and otherwise at cursor.

Display	Description
(a) ** RESET	RESET signal is input to user MPU.
(b) ** BUS REQUEST	BUS REQUEST signal is input to user MPU.
(c) ** TOUT S=x A=xxxx xxxxx	Since WAIT signal is input more than 80μS to user MPU, time out occurs. "S=x" represents the following operation state and A=xxxx xxxxx represents the address where time out occurs. Or a parity error occurs with emulation memory access.

Value of S	Operation state
0	HALT state
1 or 3	DMA operation
2	first op-code fetch
5	SLEEP state
6	second or third op-code fetch
7	CPU operation except op-code fetch

Note) In the above case, pressing the BREAK key terminates STEP command. At this time, the ASE terminates by input of $\overline{\text{RESET}}$ signal to user MPU. Therefore, note that the internal I/O register will be reset.

(7) Single-step execution of NMI servicing

If the user system is executed during NMI servicing (IEF1=0 and IEF2=1), to execute single step from the servicing, always specify N option after starting address. Without specifying IEF1 flag forcedly becomes 1 by the ASE. Then it might not be executed correctly.

Note that only IEF2 flag is displayed in the format of "IF=x" as register display. Please judge whether it is in NMI servicing or not.

For details of IEF1 or IEF2 (Interrupt Enable Flag), refer to "HD64180 User's Manual".

:STEP [<starting address>] N [;[<execution step number>][{J / N}]] CR

(8) Interactive input format

```
: STEP ? CR
START ADDRESS (RESET/addr/return) ? xx CR
NMI PROCESS (Y/N) ? xx CR
EXEC COUNT (n/return) ? xx CR
ALL, JUMP(J) OR NO(N) TRACE (ALL/J/N) ? x CR
```

Notes

(1) If the step number to be executed is not specified, single step is executed.

(2) During SLP or HALT instruction execution, the ASE cancels SLEEP or HALT with NMI. Then PC becomes $0066.

(3) As for the execution by STEP command, the following is not input to user MPU.

- $\overline{\text{RESET}}$
- $\overline{\text{NMI}}$
- $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, $\overline{\text{INT2}}$
- $\overline{\text{BUSREQ}}$
- $\overline{\text{DREQ1}}$, $\overline{\text{DREQ2}}$

Note that the internal interrupt is accepted.

(4) If an internal interrupt (timer, DMA, CSI/O or ASCI interrupt) occurs, the ASE executes only vectoring. In this case, the instruction execution is displayed as "***".

(5) The ASE does not execute DMA transfer with STEP command. However, the ASE transfers only one byte when the instruction of starting DMA (DMAE bit on) is executed.

Example

(1) When executes single step of the user program from $1000;

```
:STEP 1000 CR
PC=1002 01002 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=54=.Z*H*V..  B=00  C=00  D=28  E=00  H=20  L=00 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=05
LD    A,00
STEP NORMAL END
```

(2) When executes two steps from $1000;

```
:STEP 1000:2 CR
PC=1002 01002 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=54=.Z*H*V..  B=00  C=00  D=28  E=00  H=20  L=00 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=28
LD    A,00
PC=1003 01003 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=54=.Z*H*V..  B=00  C=00  D=28  E=00  H=20  L=00 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=4B
AND   A
STEP NORMAL END
```

(3) When not displays during sixteen-step execution from $1000;

```
:STEP 1000:10 N CR
PC=1010 01010 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=44=.Z*.*V..  B=00  C=F6  D=28  E=0A  H=20  L=0A CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=05
LDIR
STEP NORMAL END
```

(4) When displays only executing branch instruction during sixteen-step execution from $1000;

```
:STEP ? CR
START ADDRESS (RESET/addr/return) ?1000 CR
NMI PROCESS (Y/N) ?N CR
EXEC COUNT (n/return) ?10 CR
ALL,JUMP(J) OR NO(N) TRACE (ALL/J/N) ?J CR
PC=1010 01010 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=54=.Z*H*V..  B=01  C=00  D=28  E=00  H=20  L=00 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=57
JP    Z,1010
PC=1010 01010 SP=3FFF 03FFF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
 A=00  F=44=.Z*.*V..  B=00  C=F6  D=28  E=0A  H=20  L=0A CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=3F
LDIR
STEP NORMAL END
```

9.35 STOP SET, DISPLAY, AND CANCEL THE STOP CONDITIONS

Command Syntax

```
STOP (CR)
```

Function

(1) Sets, displays, and cancels conditions which stop the single step execution in the STEP command.
(2) Can set a maximum of 5 stop conditions at a once.

Description

(1) With the STOP command input, the former 1st STOP condition is displayed and the sub-command mode is activated. The specifications of the display format and sub-command are shown as follows.

```
:STOP (CR)
<Former 1st stop condition> ? [ { - | <stop condition> } ] (CR)

<Former 2nd stop condition> ? [ { . | + | * } <stop condition> ] (CR)
        :                                  :
<Former 5th stop condition> ? [ { . | + | * } <stop condition> ] (CR)
```

- : Cancels all stop conditions. Only in the case of the 1st stop condition, can be specified. With this condition input, stop command is finished.

. : Previous stop conditions just specified are set as new stop conditions.

+ <stop condition> :
OR combination conditions with previous stop condition

* <stop condition> :
AND combination conditions with previous stop condition

Only (CR) :
Former stop conditions are defined as new stop conditions.

When both "+" and "*" are in the specified condition, "*" has a priority.

(2) Setting is shown as follows.

(a) PC (program counter)

. PC { = | > | < | ≥ | ≤ } <address>

When the PC value and the value specified in <address> satisfy the conditions in { } , the single-step execution stops. If the input address mode is physical address, the address is converted to the logical address, then the ASE specifies and checks the logical address as a condition.

(b) Specification of PC range

<address 1> { ≥ | ≤ | = } . PC { ≥ | ≤ | = } <address 2>

When the PC value enters in the specified range of address 1 and 2, the single-step execution stops. The ASE checks the logical address as a condition as well as (a).

(c) Value of the register

. <register name> { = | ≥ | ≤ | = } <data>

When the specified value of the register and <data> satisfies the conditions in {△} , the single-step execution stops. (I, R, F, ZF and IF registers are excluded.)

(d) Specification of the range of the register value

<data 1> { ≥ | ≤ | = } . <register name> { ≥ | ≤ | = } <data 2>

When specified value of the register enters the specified range of data 1 and 2, the single-step execution stops.

(e) Memory contents

<address 1> { = | ≥ | ≤ | = } <address 2> [; { B | W | L }]

B: 1 byte
W: 2 bytes (Default)
L: 4 bytes

When they meet the conditions in { } , compared with address memory contents specified in addersses 1 and 2, the execution stops. Size to be compared depends on options. If the option is defaulted they are compared with two bytes. If the input address mode is logical address, the ASE calculates a physical address from the current MMU register contents every single instruction, then checks the contents.

(f) Specification of the range of memory contents

<address 1> { > / < / >= / <= } <address 2> { > / < / >= / <= } <address 3> [; { B / W / L }]

B: 1 byte
W: 2 bytes (default)
L: 4 bytes

When the address contents specified in address 2 are in the range of those specified in addresses 1 and 3, the single-step execution stops. During default, they are compared with two bytes. If the input address mode is logical address, the ASE performs in the same way as (e).

(g) Flag register contents

. { F / ZF } = <bit pattern>

When the flag register contents equal the specified bit pattern, the single-step execution stops. Bit pattern consists of up to 6 digits max., having either 0,1, or period. 0 means "OFF"; 1, "ON"; period, "Ignore".

. F=0.11.0

S: Sign bit
Z: Zero
H: Half-carry
V: Parity/Over-flow
N: Negate
C: Carry

If specify less than 6 digits, the omitted digit is regarded as a period (.).

(h) Change of flag register contents

. { F / ZF } = <bit pattern 1> > <bit pattern 2>

When the flag register contents change from bit pattern 1 to bit pattern 2, the single-step execution stops.

(3) Interactive input format

: STOP ? CR

<Former 1st stop condition> ? [{ - / <stop condition> } .] CR

<Former 2nd stop condition> ? [{ { . / + / * } <stop condition> }] CR

:

:

<Former 5th stop condition> ? [{ { . / + / * } <stop condition> }] CR

Note

(1) When the condition of memory contents is specified in logical address input mode, it always checks if the condition is satisfied with logical address even if the contents in MMU control register are changed.

Example

(1) When specifies ".PC=1251" as a stop condition of single-step emulation;

```
:STOP (CR)
                                    ? .PC=1251 (CR)
                                    ? (CR)
```

(2) When regards the time when A register exceeds $80 as a stop condition;

```
:STOP (CR)
   .PC=1251                         ? .A>=80 (CR)
                                    ? (CR)
```

(3) When cancels all stop conditions;

```
:STOP (CR)
   .A>=80                           ? - (CR)
```

(4) When regards the AND combination of (a) to (c) shown below as a stop condition;
(a) Program counter is in the range from $1000 to $1FFF.
(b) A register becomes $FF.
(c) Carry flag is 1.

```
:STOP ? (CR)
                                    ? 1000<=.PC<=1FFF (CR)
                                    ? *.A=FF (CR)
                                    ? *.F=.....1 (CR)
                                    ? (CR)
```

9.36 SYMBOL REGISTER, ELIMINATE, AND DISPLAY OF SYMBOLS

Command Syntax

```
SYMBOL  { △  <symbol name> = <address>  CR
        { △-[ <symbol name>]            CR
        {   [△<symbol name>]            CR
```

Function

(1) Registers new symbol name.
Symbol names consist of character rows made of alphanumerics up to 8 digits starting with an alphabet.
When input, "double quotation mark" is needed just before the symbol name.
(2) Eliminates one symbol name or all those which have been registered.
(3) Displays registered symbol name.

Description

(1) Register symbol name
:SYMBOL <symbol name> = <address> CR
(2) Eliminate a registered symbol
:SYMBOL-<symbol name> CR
(3) Eliminate all registered symbols
:SYMBOL - CR
(4) Display address of one registered symbol name
:SYMBOL <symbol name> CR
(5) Display address of all registered symbol name
:SYMBOL CR

```
<SYMBOL>        <ADDR>  <SYMBOL> <ADDR> ---------
<symbol name>   <address>  -------------------------
     |             |
     |             |
     |             |
<address>    xxxx xxxxx        ① Logical address
             ①    ②            ② Physical address
```

(6) Interactive input format
(a) Register

```
:SYMBOL ? CR
 DISPLAY SYMBOLS (Y/N)  ? N CR
 SET, DELETE OR ALL DELETE (S/D/ALL)  ? S CR
 SYMBOL NAME ("symbol )  ? <symbol name> CR
 ADDRESS  ? <address>CR
```

(b) Elimination

```
:SYMBOL  ? CR
 DISPLAY SYMBOLS  (Y/N)  ? N CR
 SET, DELETE OR ALL DELETE  (S/D/ALL)  ? {D}  CR
                                         {ALL}
 SYMBOL NAME ("symbol)  ? <symbol name> CR
```

In the case of inputting "ALL", "SYMBOL NAME ("symbol) ?" is not displayed.

(c) Display

```
:SYMBOL  ? (CR)
 DISPLAY SYMBOLS (Y/N) ? Y (CR)
 ALL OR ONE SYMBOL  (ALL/O) ? {ALL / O}  (CR)
 [SYMBOL NAME ("symbol) ? <symbol name> (CR) ]
```

In the case of inputting "ALL", "SYMBOL NAME ("symbol) ?" is not displayed.

Note

(1) Symbol names consist of character rows of alphanumerics up to 8 digits. The first one must be an alphabet.
(2) Symbol numbers of up to 1000 max. can be registered.
(3) Symbol holds a physical address. Therefore, the specified address in logical address input mode is converted the physical address to be registered. Also, when a symbol name is referred in logical address input mode, the ASE uses the converted address from holding physical address to a logical address. In this case, if a logical address becomes indefinite, an error occurs.
(4) If the user tries to register the symbol name already exists, later specified symbol name is valid. The address the symbol name has will be changed.

Example

(1) When registers $1261 to symbol name "SAMPLE1";

```
:SYMBOL "SAMPLE1=1261 (CR)
```

(2) When registers $1245 to symbol name "SAMPLE2" in interactive input format;

```
:SYMBOL ? (CR)
 DISPLAY SYMBOLS (Y/N) ?N (CR)
 SET,DELETE OR ALL DELETE (S/D/ALL) ?S (CR)
 SYMBOL NAME ("symbol) ?"SAMPLE2 (CR)
 ADDRESS ?1245 (CR)
```

(3) When displays all registered symbols (interactive input format);

```
:SYMBOL ? (CR)
 DISPLAY SYMBOLS (Y/N) ?Y (CR)
 ALL OR ONE SYMBOL (ALL/O) ?ALL (CR)
 <SYMBOL> < ADDRESS >   <SYMBOL> < ADDRESS >   <SYMBOL> < ADDRESS >
 SAMPLE1   1261 01261   SAMPLE2   1245 01245
```

(4) When eliminates all registered symbols (interactive input foramt);

```
:SYMBOL ? (CR)
 DISPLAY SYMBOLS (Y/N) ?N (CR)
 SET,DELETE OR ALL DELETE (S/D/ALL) ?ALL (CR)
```

9.37 T R A C E | DISPLAY OF TRACE BUFFER CONTENTS

Command Syntax

```
TRACE [△[[[-] <n>]][;B]] (CR)
  n : Number of display instructions or display cycles
  B : display every bus cycle unit
```

Function

(1) Fetches only an instruction out of the trace buffer and displays in instruction mnemonic.
(2) Trace buffer contents are displayed by machine cycle units. In this case if DMA transfer cycle, and refresh cycle are contained, they are also displayed. Refer to "5.3 Trace" for trace timing.
(3) Displays the the contents at high-speed trace.
(4) The number of instructions or cycles to be displayed can be specified.

Description

(1) In the case of displaying only the instruction mnemonic code, display the following format.

```
  LA    OP      OPRAND
  ①     ②        ③
 xxxx  xxxx     xx ~ xx
```

① Instruction address (display only logical address)
② Mnemonic instruction
③ Instruction operand

(2) When displaying by machine cycle, display the following format.

```
 LA   OP   OPRAND   AB    DB  R/W  ST  E  NMI INT RES  DREQ  TEND  PROB
 ①    ②      ③      ④     ⑤   ⑥   ⑦  ⑧  ⑨   ⑩   ⑪    ⑫     ⑬     ⑭
xxxx xxxx xx ~ xx  xxxxx xx  xx    x  xx  x   x   x     xx    xx    xxxxxxxx
```

① Logical address
② Mnemonic instruction
DMA transfer cycle and refresh cycle execute the next display.

"*DMA*" ----- DMA transfer cycle
"*REF*" -- Refresh cycle

③ Instruction operand
④ Address bus contents (physical address)
The value of address bus (A0 to A18) is displayed.
⑤ Data bus contents (D0 to D7)
⑥ RD and WR signal
"R" means RD signal is "Low".
"W" means WR signal is "Low".
"-" means RD and WR signals are "High".

⑦ Operation states

Each signal states of ST, HALT, and LIR and the operation states of MPU are shown.

Table 9-3 Operation state

Display	ST	$\overline{\text{HALT}}$	$\overline{\text{LIR}}$	Operation status
0	0	0	0	HALT mode
1	0	0	1	DMA operation
2	0	1	0	CPU operation (1st op-code fetch)
3	0	1	1	DMA operation
4	1	0	0	Not applied
5	1	0	1	SLEEP mode
6	1	1	0	CPU operation (2nd or 3rd op-code fetch)
7	1	1	1	CPU operation (except op-code fetch)

⑧ $\overline{\text{ME}}$ and $\overline{\text{IOE}}$ signals

"M" --------- $\overline{\text{ME}}$ signal is "Low".
"I" --------- $\overline{\text{IOE}}$ signal is "Low".
"-" --------- ME and IOE signals are "High".

⑨ $\overline{\text{NMI}}$ signal

"0" --------- $\overline{\text{NMI}}$ signal is "Low".
"1" --------- $\overline{\text{NMI}}$ signal is "High".

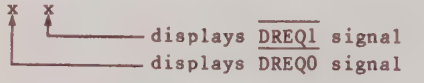
⑩ $\overline{\text{INT0}}$, $\overline{\text{INT1}}$, and $\overline{\text{INT2}}$ signals

"0" -------- $\overline{\text{INTn}}$ signal is "Low".
"1" -------- INTn signal is "High".

⑪ $\overline{\text{RESET}}$ signal

"0" -------- $\overline{\text{RESET}}$ signal is "Low".
"1" -------- RESET signal is "High".

⑫ $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ signals



"0" ------- $\overline{\text{DREQn}}$ signal is "Low".
"1" ------- $\overline{\text{DREQn}}$ signal is "High".

⑬ $\overline{\text{TEND0}}$ and $\overline{\text{TEND1}}$ signals

```
x x
↑ └──── displays TEND1 signal
└────── displays TEND0 signal
```

"0" -------- $\overline{\text{TENDn}}$ signal is "Low".
"1" -------- $\overline{\text{TENDn}}$ signal is "High".

⑭ Signal level of 8 external probes
0 - Low
1 - High
Displayed in order of No.1 to No.8.

(3) After execution in high-speed trace mode, display is the following format.

CYCLE	LA	AB	DB	R/W	ST	E	NMI	INT	RES	DREQ	TEND	PROB
①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	⑪	⑫	⑬
-&xxxx	xxxx	xxxxx	xx	xx	x	xx	x	x	x	xx	xx	xxxxxxxx

① How many cycles remain before execution stop is displayed in decimal
② The same as (2) ①
③ to ⑬ The same as (2) ④ to ⑭

(4) Display number specification after execution in high-speed trace mode is read as the cycle number of 125nS unit. The display number specification after execution in other mode is read as instruction number.
:TRACE n CR

(5) Right after execution of GO or STEP command, the trace buffer pointer locates the last trace information. Specify display number with "-" until the pointer moves with TRACE_POINTER command.
:TRACE -n CR

(6) In the case of displaying all contents of the trace buffer, specify them following the next format.

```
:TRACE (CR)
  or
:TRACE ;B (CR)          Display every bus cycle
```

(7) Interactive input format

(a) Instruction trace

```
:TRACE ? (CR)
 NUMBER OF DISPLAY INSTRUCTIONS (ALL/n)  ? x (CR)
[NUMBER IS MINUS (Y/N)  ?  x (CR) ] ----(not displayed at ALL
                                         specification)
 BUS CYCLE INFORMATION (Y/N)  ?  x (CR)
```

(b) High-speed trace

```
:TRACE ? (CR)
 NUMBER OF DISPLAY CYCLES  (ALL/n)  ? x (CR)
[NUMBER IS MINUS (Y/N)  ?  x (CR) ]  ----(not displayed at ALL
                                         specification)
```

Note

(1) Ignore B option after execution with the high-speed trace mode.
(2) Use TRACE_POINTER command to move the trace buffer pointer.
(3) In the case of all data display, regardless of trace buffer pointer position, the buffer is displayed from the start.
(4) When the display numbers are larger than those to be displayed, specifying "-" causes to display from the head of the buffer to the current point and, without specifying "-", display from the current point to the end.
(5) Trace buffer contents are kept until next GO or STEP command is executed.
(6) Data up to the most recent 2048 cycles during operation are buffered in the trace buffer. In the case of data of less than 2048 cycles, only the data of operated cycles are buffered.
(7) When specifying the trace save range with TRACE_BREAK_CONDITION command, there might be no program fetch. In this case, if there's no trace information of the first op-code fetch, the number of the specified display instruction is displayed as a number of bus cycles.
 (Ex.)
 :T -5 (CR) When there is no first op-code in trace buffer in specifying this, the ASE displays the last 5 bus cycles of trace information.
(8) "-" does not mean a negative value. It means a direction to move a pointer backward.

(9) When using timer function, A18 pin is regarded as Tout pin. The ASE does not save Tout information as trace information. In this case, A18 is displayed as 0.

(10) Each of the following two instructions is three byte long. When the condition is not satisfied, the MPU traces the next instruction cycle without operand fetch of the third byte (high-order byte of operand to be jumped).

```
JP      f, mn
CALL    f, mn     f(condition code)=NZ, Z, NC, C, PO, PE, P, M
                  mn=branch address
```

In this case, the complete address to be jumped cannot be displayed from the trace buffer. So, "xxxx" is displayed as the address to be jumped the TRACE command.

(Ex.) Execution instruction

```
1000   XOR A
1001   JP  NZ, 2000  --- In this case, since the condition is not
1004   LD  HL, 2800      satisfied (Z flag=1), the jump to $2000
                         does not occur.
```

Trace display after execution

```
1000   XOR A
1001   JP  NZ, xxxx
1004   LD  HL, 2800
```

(11) Some ASE informations are saved into the trace buffer when user program execution stop. In this case, the ASE program uses 1 to 20 cycles for emulation stop. TRACE command displays "*ASE*" for this cycle at high-speed trace or at bus cycle trace with B option and display "**" for data bus because data is indefinite.

(Ex.)

```
:TRACE  -1;B
   LA    OP    OPRAND   AB    DB R/W ST E NMI INT RES DREQ TEND   PROB
 1006  INC   HL       01006 23   R   2 M  1  111  1   11   11  11111111
 1007         *ASE*    01007 **   R   2 M  1  111  1   11   11  11111111
```

Also, in case of the tarce range condition is specified, and user program execution stop out of the trace range, this ASE information is saved into the trace buffer.

Example

(1) When specifies the break condition at $1006, and displays all the contents saved in the trace buffer in instruction code after executing the user program from $1000;

```
:TRACE (CR)
  LA    OP        OPRAND
1000  LD      HL,1200
1003  CALL    1100
1100  LD      A,(HL)
1101  INC     A
1102  AND     A
1103  LD      (HL),A
1104  RET
1006  INC     HL
```

(2) When specifies the display number for 6 instructions(6 instructions backward from the last saved instruction) in the same condition as example (1), and displays in bus cycle unit;

```
:TRACE -6;8 (CR)
  LA    OP        OPRAND          AB     DB  R/W  ST  E  NMI  INT  RES  DREQ  TEND   PROB
1100  LD      A,(HL)          01100  7E   R    2   M    1   111   1    11    11   11111111
1200                          01200  00   R    7   M    1   111   1    11    11   11111111
1101  INC     A               01101  3C   R    2   M    1   111   1    11    11   11111111
1102  AND     A               01102  A7   R    2   M    1   111   1    11    11   11111111
1103  LD      (HL),A          01103  77   R    2   M    1   111   1    11    11   11111111
1200                          01200  01   W    7   M    1   111   1    11    11   11111111
1104  RET                     01104  C9   R    2   M    1   111   1    11    11   11111111
1FFD                          01FFD  06   R    7   M    1   111   1    11    11   11111111
1FFE                          01FFE  10   R    7   M    1   111   1    11    11   11111111
1006  INC     HL              01006  23   R    2   M    1   111   1    11    11   11111111
1007                  *ASE*   01007  **   R    2   M    1   111   1    11    11   11111111
```

(3) When displays user program 32 cycles before execution stop, after executing in high-speed trace mode (interactive input format);

```
:GO 1000;H (CR)
 PROCESS STARTED FROM 1000 01000
:TRACE ? (CR)
NUMBER OF DISPLAY C   ES (ALL/n) 720 (CR)
NUMBER IS MINUS (Y/   'Y (CR)
 CYCLE   LA    AB     DB  R/W  ST  E  NMI  INT  RES  DREQ  TEND  REF   PROB
-$0032  1FFE  01FFE  10   R    7   M    1   111   1    11    11    1   11111111
-$0031  1FFE  01FFE  10   R    7   M    1   111   1    11    11    1   11111111
-$0030  1FFE  01FFE  10   -    7   -    1   111   1    11    11    1   11111111
-$0029  00AC  000AC  FF   -    7   M    1   111   1    11    11    0   11111111
-$0028  00AC  000AC  FF   -    7   M    1   111   1    11    11    0   11111111
-$0027  00AC  000AC  FF   -    7   M    1   111   1    11    11    0   11111111
-$0026  10AE  010AE  FF   -    7   -    1   111   1    11    11    1   11111111
-$0025  1006  01006  FF   R    2   M    1   111   1    11    11    1   11111111
-$0024  1006  01006  23   R    2   M    1   111   1    11    11    1   11111111
-$0023  1006  01006  23   R    2   M    1   111   1    11    11    1   11111111
-$0022  1006  01006  23   R    2   M    1   111   1    11    11    1   11111111
-$0021  1006  01006  23   R    2   M    1   111   1    11    11    1   11111111
-$0020  1006  01006  23   R    2   M    1   111   1    11    11    1   11111111
-$0019  1006  01006  23   R    2   M    1   111   1    11    11    1   11111111
-$0018  00AD  000AD  23   -    7   -    1   111   1    11    11    0   11111111
-$0017  00AD  000AD  FF   -    7   M    1   111   1    11    11    0   11111111
-$0016  00AD  000AD  FF   -    7   M    1   111   1    11    11    0   11111111
-$0015  00AD  000AD  FF   -    7   -    1   111   1    11    11    0   11111111
-$0014  00AD  000AD  FF   -    7   -    1   111   1    11    11    1   11111111
-$0013  10AF  010AF  FF   -    7   -    1   111   1    11    11    1   11111111
-$0012  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0011  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0010  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0009  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0008  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0007  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0006  1007  01007  **   R    2   M    1   111   1    11    11    1   11111111 *ASE*
-$0005  00AE  000AE  FF   -    7   -    1   111   1    11    11    0   11111111
-$0004  00AE  000AE  FF   -    7   M    1   111   1    11    11    0   11111111
-$0003  00AE  000AE  FF   -    7   M    1   111   1    11    11    0   11111111
-$0002  00AE  000AE  FF   -    7   -    1   111   1    11    11    0   11111111
-$0001  1008  01008  FF   -    6   -    1   111   1    11    11    1   11111111
```

9.38 TRACE_BREAK_CONDITION — SET, DISPLAY, CANCEL EXECUTION CONDITIONS

Command Syntax

```
TRACE_ BREAK_ CONDITION (CR)
TRACE START CONDITIONS   <former trace start condition >
? [<new trace start condition >] (CR)
TRACE RANGE CONDITIONS   <former trace save condition >
? [<new trace save condition >]  (CR)
BREAK CONDITIONS 1       <former break condition  1>
? [<new break condition >]  (CR)
BREAK CONDITIONS 2       <former break condition  2>
? [<new break condition >]  (CR)
```

Functions

(1) Sets, displays, and cancels the break condition, trace start condition, and trace save condition in user program execution by the GO command. Refer to "5.2 Break" for break timing.

Description

(1) Trace start condition ----- Conditions that cause the start of saving run data into the trace buffer. Execution terminate when the buffer become full. Also, when another condition has been already specified, the condition is cancelled.

Note)
When specifying the trigger mode, high-speed trace mode, and cycle reset mode with the GO command, <trace start condition > is ignored.

(2) Trace save condition
To save the execution data in the trace buffer. Only when they are satisfied, data is saved in the trace buffer.

Note)
(a) This condition is not set with <trace start condition> at the same time. If <trace start condition> is set, <trace save condition> is automatically cancelled.

(b) When specifying the trigger mode, high-speed trace mode, and cycle reset mode with the GO command, <trace save condition> is ignored.

(3) Break condition 1 / Break condition 2 ------ Conditions which break the execution. Two conditions can be selected and the execution breaks when either of them is satisfied.

Note)
(a) When specifying trigger mode in the GO command
When this break condition is satisfied, the trigger pulse is output from the trigger pin so as not to break the program.
(b) When specifying <trace save condition >, <break condition 2> can not be set.
(c) When selecting the count conditions in <break condition 1>, <break condition 2> cannot be set.

(4) Conditions to be specified are shown as follows.

(a) PC (Program counter)

.PC = <address>

When executing specified address instructions, conditions are satisfied. The ASE checks only when both LIR and RD signals are "Low". This point is different from specifying by MA.

(b) Data

DATA = <data> [;{R / W}]

Data : 1-byte data
R : Check data at data READ
W : Check data at data WRITE
No specification of R,W : Check data at data READ/WRITE

When selected data is accessed under the READ/WRITE conditions, the conditions are satisfied.

(c) Address

MA=<address> [;<option>]
Address : address value
Option : I ------ Check when access of I/O space
R ------ Check when READ of memory space
W ------ Check when WRITE of memory space
IR ------ Check when READ of I/O space
IW ------ Check when WRITE of I/O space
No specification
------ Check the address access regardless of space types

When the specified address is accessed accroding to the specified option, the conditions are satisfied.

(d) External probe

PROB = a1 a2 a3 a4 a5 a6 a7 a8

an = 0 : When signal of the probe No. n is "0", the conditions are satisfied.
an = 1 : When signal of the probe No. n is "1", the conditions are satisfied.
an = . : Signal of the probe No. n is ignored.

When all specified probe signals are met, the conditions are satisfied. When specifies after the probe No. n are all periods (Ignore probe contents), you can omit that data input.

(e) Specification of range

(i) Specification of PC range

<address> { > | >= | < | <= } .PC { > | >= | < | <= } <address>

(ii) Specification of address range

<address> { > | >= | < | <= } MA { > | >= | < | <= } <address> [;<option>]

Option:

I	------ Check when access of I/O area
R	------ Check when READ of memory area
W	------ Check when WRITE of memory area
IR	------ Check when READ of I/O area
IW	------ Check when WRITE of I/O area
No specification	----- Check the address range access regardless of area

When the specified address is accessed according to the specified option, the conditions are satisfied.

Note)

At the range specification, the condition of 2^0-bit of address bus cannot be specified.

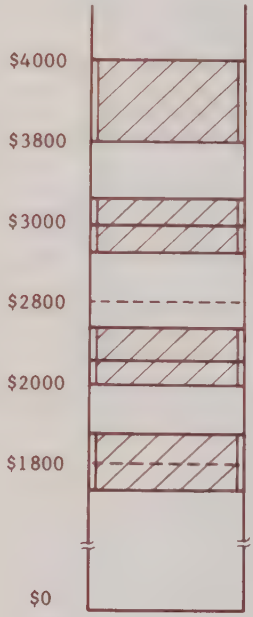
(Ex.) Range specification

specification	satisfied condition
1 < MA < 5	2 <= MA <= 5
1 <= MA < 5	0 <= MA <= 5
1 < MA <= 5	2 <= MA <= 5
1 <= MA <= 5	0 <= MA <= 5
2 < MA < 6	2 <= MA <= 5
2 <= MA < 6	2 <= MA <= 5
2 < MA <= 6	2 <= MA <= 7
2 <= MA <= 6	2 <= MA <= 7

Note) PC and address can be specified only when their range is from $000 to $7FF or $800 to $FFF (2K bytes max.) on 3 low-order digits and the same on high-order digits.

The range cannot be specified if it exceeds the boundary $x000 or $x800 (x=0,1,2,...) every 2K bytes on the memory.

(Ex.)



Range specification

$3800 <= .PC <= $3FFF
Specified correctly.

$2E00 <= .PC <= $32FF
Wrong because crosses $3000 boundary.

$23C0 <= .PC <= $2640
Specified correctly.

$1600 <= .PC <= $2ABF
Wrong because crosses $1800 boundary.

(f) TEND signal

$$\begin{Bmatrix}\text{TEND0}\\ \text{TEND1}\end{Bmatrix}$$

The TEND0 and TEND1 are transfer end signals of HD64180 internal DMAC. When this signal becomes the low level, the conditions are satisfied.

(g) Trace save except the specified conditions (can be specified only under the trace save conditions)

DISABLE

This can be specified only under <trace save condition> and must be combined with other condition. With this selection, the execution contents, except the specified conditions, are traced.

(h) Times of specification of conditions

COUNT = <times> times : 1 to 1024

After the specified conditions satisfy these specification times, the condition is satisfied. In the case that it is specified with <trace start condition>, when the specification times are satisfied, start the trace save. In the case that they are specified with <break condition 1> when the specification times are satisfied, the GO command breaks. This condition cannot be specified with <trace start condition> and <break condition 2>. When you specify this in <break condition 1>, <break condition 2> cannot be specified. This condition uses combination form.

(i) Combination

Each condition of the above-mentioned (a) to (h) can be combined and selected. Each condition is connected with "*". Maximum combinations are shown as follows. Possible combinations of conditions are shown in table 9-5 in "Note".

(5) Cancellation of each condition

(a) Cancellation of the conditions in display and after it

. CR

Cancels the condition in display and after it by inputting "." in specifying each condition.

(b) Cancellation of the conditions in display

- CR

Cancels the condition in display by inputting "-" in specifying each condition.

(6) When the condition is not changed,

CR

Use the former condition as it is.

(7) Interactive input format

```
:TRACE BREAK CONDITION  ? CR          <former trace start condition>
?[<new trace start condition>] CR
TRACE RANGE CONDITION                 <former trace save condition>
?[<new trace save condition>] CR
BREAK CONDITION 1                     <former break condition 1>
?[<new break condition 1>] CR
BREAK CONDITION 2                     <former break condition 2>
?[<new break condition 2>] CR
```

Note

(1) Table 9-4 shows possible combination of trace start condition, trace save condition, break conditions 1 and 2.

Table 9-4 Combination between Conditions

Condition / Combination	Trace start condition	Trace save condition	Break condition 1	Break condition 2
1	o	x	x	x
2	x	o	o	x
3	x	x	o	o

o: Possible setting
x: Impossible setting

(2) Table 9-5 shows combinations to be specified in each condition such as trace start condition, trace condition, break conditions 1 and 2.

Table 9-5 Combinations in Each Condition

Condition / Combination	PC*	DATA	MA*	PROB	TEND0 TEND1	DISABLE	COUNT	Condition to be specified
1	o	o	x	o	x	x	o	Trace start, break condition 1
2	x	o	o	o	x	x	o	Trace start, break condition 1
3	o	o	x	o	x	o	x	Trace save condition
4	o	o	x	o	x	x	x	All conditions
5	x	x	x	x	o	x	x	All conditions
6	x	x	x	x	o	x	o	Trace start, break condition 1

o: Possible setting
x: Impossible setting

*: Range specification included.

(3) If the trace start condition is specified, the ASE saves the refresh cycle trace information only with DISABLE specified.

(4) When break condition 1 or 2 is specified to breakpoint, the breakpoint has priority to be processed.

(5) In the case setting a breakpoint at SLP or HALT instruction and executing these instructions, the ASE breaks the user program by inputting NMI. Then PC becomes $0066.

(6) When a break condition is satisfied during DMA transfer (burst mode), the ASE doesn't break immediately, but after DMA transfer the ASE executes the first instruction and breaks. Note that when specifies the break condition 1 or 2 to DMA starting (DMAE bit on) instruction, the ASE transfers 1-byte data and stops.

(7) When a break condition is satisfied during DMA transfer (cycle steal mode) DMA stops.

(8) When the input address mode is logical address, the address included in the specified condition checks with logical address if the condition is satisfied, but not check with physical address. Even if change the address mode to physical address, the specified condition (incluse PC) is not change to physical address.

(9) In the case of specifying PC, the ASE checks when $\overline{LIR}$ and $\overline{RD}$ signals are both "Low". So, note that the condition is satisfied at the first, second or third op-code fetch. Especially, when using PC at trace save condition, there is no use because operand of the instruction isn't saved.

Example

(1) When specifies the memory accessed at $1000 as a trace start condition;

```
:TRACE_BREAK_CONDITION (CR)
TRACE START CONDITIONS
? MA=1000 (CR)
```

(2) When specifies the memory written into $1255 as a trace save condition;

```
:TRACE_BREAK_CONDITION (CR)
TRACE START CONDITIONS
? (CR)
TRACE RANGE CONDITIONS
? MA=1255;W (CR)
BREAK CONDITIONS 1
? (CR)
```

(3) When cancels all conditions specified with TRACE_BREAK_CONDITION;

```
:TRACE_BREAK_CONDITION (CR)
TRACE START CONDITIONS
? - (CR)
TRACE RANGE CONDITIONS MA=1255;W
? - (CR)
BREAK CONDITIONS 1
? - (CR)
BREAK CONDITIONS 2
? - (CR)
```

(4) When specifies the forth writing the data at $7F into $8201 as a break condition;

```
:TRACE_BREAK_CONDITION  CR
TRACE START CONDITIONS
? CR
TRACE RANGE CONDITIONS
? CR
BREAK CONDITIONS 1
? MA=8201;W*DATA=7F*COUNT=4 CR
```

(5) When specifies probe (No.1) 0 during the user program execution from $1800 to $1FE0 as a break condition;

```
:TRACE_BREAK_CONDITION ? CR
TRACE START CONDITIONS
? CR
TRACE RANGE CONDITIONS
? CR
BREAK CONDITIONS 1      MA=8201;W*DATA=7F*COUNT=4
? 1800<=PC<1FE0*PROB=0..... CR
BREAK CONDITIONS 2
? CR
```

9.39 TRACE_POINTER — MOVE AND DISPLAY OF TRACE BUFFER POINTER

```
TRACE _ POINTER [△{ TOP  }]  CR
                 { LAST }
                 { [-] n}
```

TOP : Moves the pointer in the top of trace buffer.
LAST: Moves the pointer in the end of trace buffer.
[-]n: Moves the pointer by [-]n.

Function

(1) Moves the display pointer on the trace buffer in which data is stored during execution of the user system.
(2) Displays the current pointer value.

Description

(1) Moves the pointer in the top of trace buffer.
:TRACE_POINTER TOP CR
(2) Moves the pointer in the end of trace buffer.
:TRACE_POINTER LAST CR
(3) In the case of moving the buffer pointer n instructions up or down after specifying and executing any option except "H" with GO command.

: TRACE _ POINTER n CR
or
: TRACE _ POINTER -n CR

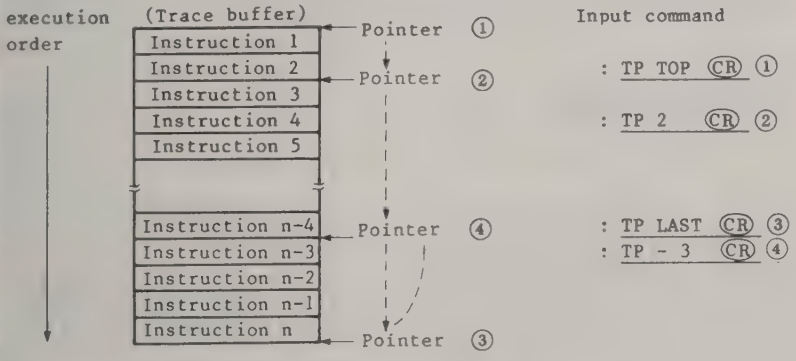


Fig. 9-13 Pointer Move of Trace Buffer (Bus cycle trace)

Note)
In the case that there is no instruction fetch in the buffer at all (for example: when trace save condition is specified at the data area access.), this moves every bus cycle in the some way as (4).

(4) In the case of moving the buffer pointer n cycles up or down after a user program execution with high-speed trace mode (GO command with H option) .

: TRACE _ POINTER n (CR)

or

: TRACE _ POINTER -n (CR)

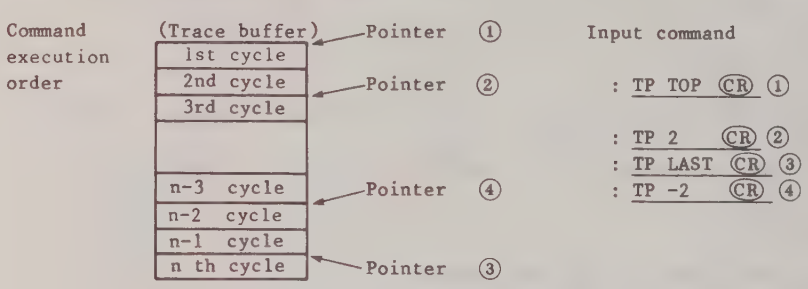


Fig. 9-13 Pointer Move of Trace Buffer (High-speed trace)

(5) In the case of display of the current pointer

: TRACE _ POINTER (CR)

TRACE POINTER=<pointer>

(6) Interactive input format

(a) Set

: TRACE _ POINTER ? (CR)

SET OR DISPLAY (S/D) ? S (CR)

SET TRACE BUFFER POINTER (TOP/LAST/n) ? x (CR)

(b) Display

: TRACE _ POINTER ? (CR)

SET OR DISPLAY (S/D) ? D (CR)

Notes

(1) 2048 cycles of trace data saved finally are buffered in the trace buffer. In the case of high-speed trace mode execution, the unit is 125nS. In another case, the unit is bus cycle. However, if cycles are saved less than these cycles or if executes with specifying trace save condition, the data is saved less than 2048 cycles.

(2) This pointer is defined by 1st TRACE command after GO or STEP command execution, so if TRACE_POINTER command is input before TRACE command, the pointer value is ignored.

(3) When larger values than the saved buffer are specified, the pointer moves to the top specified with "-" and to the end specified without "-".

Example

(1) When moves the trace buffer pointer to the top of the trace buffer;

```
:TRACE_POINTER TOP (CR)
:TRACE_POINTER (CR)
TRACE POINTER=&1 (TOP)
```

(2) When moves the trace buffer pointer three lines toward the newer data of the trace buffer;

```
:TRACE_POINTER 3 (CR)
:TRACE_POINTER (CR)
TRACE POINTER=&4
```

(3) When moves the trace buffer pointer to the newest data of the trace buffer;

```
:TRACE_POINTER ? (CR)
SET OR DISPLAY (S/D) ?S (CR)
SET TRACE BUFFER POINTER (TOP/LAST/n) ?LAST (CR)
:TRACE_POINTER ? (CR)
SET OR DISPLAY (S/D) ?D (CR)
TRACE POINTER=&221 (LAST)
```

9.40 VERIFY VERIFICATION OF MEMORY CONTENTS

Command Syntax

```
VERIFY { <file name>                                  } CR
       { [△[<value>]] [; {S} ] ] }
                         {H}
```

<value>-- address addition value
S -- S type format
H -- INTELLEC HEX format (default)

Function

(1) Compares the user program on the host system with contents on the user memory and displays the contents if they are different.

Description

(1) When connecting the SD200 CP/M-68K host system

As described in LOAD command explanation, the SD200 HD64180 cross macro assembler prepares an object program consisting of multiple files. In the VERIFY command, the following operation is performed just as in the LOAD command.

(a) When verifying all files, omit the subfile name. The ASE changes the subfile name sequentially and verifies all files except the symbol file.

(Ex.)

```
:VERIFY  B:F1 CR
```

Verify the files F1.000, F1.001 ---------- F1.007, sequentially.

(b) When verifying only one specified file, specify the subfile name.

```
:VERIFY  B:  F1.006 CR
```

Verify only the file F1.006.

(2) When connecting the general purpose host system

The object program format which the ASE loads is the program of addresses $0 to $FFFF as shown in Appendix A. To load at an optional address of 512KB in HD64180, you have to input the value to be added to the location address in the object program with LOAD command. With VERIFY command, specify the value as an operand to be added to the location address in the object program.

```
:VERIFY   [<value> ] [; { S } ]   (CR)
                        { H }
```

<value> -- Value to be added to the address data in the object program.
If nothing typed, 0 is input.

S ------- S-type format
H ------- INTELLEC HEX format (default)

Input this command and transmit data from the host system.

(3) When the object program is different from the user memory, the ASE compares them up to the last file with displaying the following messages.

```
< ADDR >        <MEM>     <FILE>
   ①             ②         ③
xxxx xxxxx       xx        xx
   :             :         :
   :             :         :
```

① Address in which a verification error occurs.
(Logical address, physical address)
② User memory contents
③ Object program contents of the host system

(4) When the verification is completed, the verified starting and end addresses are displayed.

```
TOP ADDRESS : xxxx xxxxx
END ADDRESS : xxxx xxxxx
              ①    ②
```

① Logical address
② Physical address

(5) Interactive input format

(a) In the case of SD200 CP/M-68K host system

```
:VERIFY ? (CR)
 FILE NAME  ?  <file name> (CR)
```

(b) In the case of general purpose host system

```
:VERIFY ? (CR)
 ADDITIONAL ADDRESS (addr/return)  [< value >] (CR)
 FORMAT IS S-TYPE OR HEX-TYPE  (S/H) ? x  (CR)
```

Note

(1) Displayed <top address> shows the lowest value address and <end address> displays the highest value address.

(2) For details of the conncetion method with the host system, refer to section 4.3.

(3) Take notice of the following points except in the case that the host system is SD200.

(a) Error messages are displayed when an error occurs during the transmition of data, and the VERIFY command is end.
In this case, it does not receive the remaining data transferred out of the host system. In the case of check sum error, the next error messages are output and the operation for replying the response is done.

```
CHECK SUM ERROR      ADDR=xxxx
                          (2)
CONTINUE (Y/N) ?  (1)   (CR)
```

(1) -In the case of "Y", ignore the error and proceed with the operation. In the case of "N", interrupt the operation and the VERIFY command is end.
(2) -Values at the address of the record in which an error occurs.

(b) In the case that the host system is not connected, or data is not output from the host system, the system is in the state of no reply.
In this case, pressing a BREAK key interrupts the processing and returns the ASE command input wait state.

Example

(1) When verifies all object files having "R:OBJF" as a file name in connecting to SD200 CP/M-68K host system (note that a object file is prepared by CP/M-68K 64180 cross macro assembler);

```
:VERIFY B:OBJF (CR)
TOP ADDRESS : 0000 00000
END ADDRESS : E234 7F234
```

(2) When verifies only the specified file under the same condition as example (1) above;

```
:VERIFY ? (CR)
FILE NAME ?B:SAMPLE.LO (CR)
TOP ADDRESS : 1000 01000
END ADDRESS : 17FF 017FF
```

(3) When verifies in S-type format in connecting to a general purpose host system;

```
:VERIFY ;S (CR)
```

(4) When verifies in INTELLEC HEX format in connecting to a general purpose host system (interactive input format);

```
:VERIFY ? (CR)
ADDITIONAL ADDRESS (ADDR/RETURN) ? (CR)
FORMAT IS S-TYPE OR HEX-TYPE (S/H) ?H (CR)
```

9.41 .<register name> — REFER AND CHANGE THE REGISTER

Command Syntax

```
.<register name> [△<value> ] (CR)

Register name
          .A .B .C .D .E .H .L .F          (register set GR)
          .ZA .ZB .ZC .ZD .ZE .ZH .ZL .ZF  (register set GR')
          .PC .SP .IX .IY .I .R
```

Function

(1) Sets the specified value in the register.
(2) Displays the register contents and requests a change. When the change value is specified, it is set in the register and displays the next register contents and requested a change.
This operation is repeated following the . A to .ZF order.

Description

(1) In the case of changing the specified register contents

```
: . <register name>  <value>  (CR)
```

(2) In the case of changing each register contents referring the register contents

```
: . A (CR)
  . A   =  xx ? <①>(CR)
  . B   =  xx ? <①>(CR)
  . C   =  xx ? <①>(CR)
  . D   =  xx ? <①>(CR)
  . E   =  xx ? <①>(CR)
  . H   =  xx ? <①>(CR)
  . L   =  xx ? <①>(CR)
  . F   =  xx = SZ*H*VNC ? <①>(CR)
  . PC  =  xxxx xxxxx ? <①>(CR)
  . SP  =  xxxx xxxxx ? <①>(CR)
  . IX  =  xxxx ? <①>(CR)
  . IY  =  xxxx ? <①>(CR)
  . I   =  xx ? <①>(CR)
  . R   =  xx ? <①>(CR)
  . ZA  =  xx ? <①>(CR)
  . ZB  =  xx ? <①>(CR)
  . ZC  =  xx ? <①>(CR)
  . ZD  =  xx ? <①>(CR)
  . ZE  =  xx ? <①>(CR)
  . ZH  =  xx ? <①>(CR)
  . ZL  =  xx ? <①>(CR)
  . ZF  =  xx = SZ*H*VNC ? <①>(CR)
```

① Specifies either

<value>	This value is set in the register.
<period>	Terminate this command.
<no selection>	The next register contents are displayed.

The order of register contents display has been explained before. If the command is input to the other registers than .A register, the ASE does not display the contents of registers before the specified registers After .ZF register display, the ASE goes to command input wait state.

(Ex.):

```
. ZH CR

.ZH = xx ? CR
.ZL = xx ? CR
.ZF = xx = SZ*H*VNC ? CR
:
```

(3) Interactive input format

```
:. A ? CR
. A = xx ? <①>CR
. B = xx ? <①>CR
  :   :
  :   :
  :   :
.ZF = xx = SZ*H*VNC ? <①>CR
```

① The same as (2).

Note

The ASE cannot change two bytes at the same time as 16-bit register on register pair BC, DE and HL. So, always change every 1-byte register (such as B register and C register).

Example

(1) When displays from A register of MPU in order;

```
:.A   (CR)
 .A =02 ?  (CR)
 .B =00 ?  (CR)
 .C =F6 ?  (CR)
 .D =28 ?  (CR)
 .E =0A ?  (CR)
 .H =12 ?  (CR)
 .L =01 ?  (CR)
 .F =10=..*H*... ? (CR)
 .PC=1007 01007 ?  (CR)
 .SP=1FFF 01FFF ?  (CR)
 .IX=0000 ?  (CR)
 .IY=0000 ?  (CR)
 .I =00 ?  (CR)
 .R =27 ?  (CR)
 .ZA=00 ?  (CR)
 .ZB=00 ?  (CR)
 .ZC=00 ?  (CR)
 .ZD=00 ?  (CR)
 .ZE=00 ?  (CR)
 .ZH=00 ?  (CR)
 .ZL=00 ?  (CR)
 .ZF=00=..*.*... ? (CR)
```

(2) When sets $5C66 to program counter (.PC), $E9FF to stack pointer (.SP), and $FF to A register (.A), then displays all registers;

```
:.PC 5C66  (CR)
:.SP E9FF  (CR)
:.A FF     (CR)
:REGISTER  (CR)
PC=5C66 05C66 SP=E9FF 0E9FF IX=0000 IY=0000 IR=00 C1LA=F000 BKLA=0000 IF=0
  A=FF  F=10=..*H*...  B=00  C=F6  D=28  E=0A  H=12  L=01 CMBR=00000 I=00
ZA=00 ZF=00=..*.*... ZB=00 ZC=00 ZD=00 ZE=00 ZH=00 ZL=00 BKBR=00000 R=27
```

Table 10-1 Error Messages

No.	Error Message	Description/Solution
1	INTERNAL ERROR (nn)	ASE software involves an error. Please make contact with the Hitachi agency. The number in parenthesis shows the error cause.
2	ASE HARDWARE ERROR	ASE hardware involves an error. Please make contact with the above-mentioned agency. The message displayed just before this shows the error cause.
3	HOST ABNORMAL END	The interface program on SD200 ended abnormally. Restart it.
4	ABNORMAL USER LINE	The signal input into the 64-pin connector between the user system and the ASE is incorrect. Examine the incorrect pins by CHECK command.
5	USER SYSTEM NOT READY	The CLOCK signal has not yet come from the user system. Clock is changed to the ASE 6.144MHz. Examine the reason for CLOCK not coming and restart it.
6	HOST NOT READY	The ASE program on SD200 is not activated by HOST command yet. Activate with HOST command.
7	PRINTER NOT READY	The printer does not operate because of no-connection or power OFF. The ASE stops the printer and continues outputting on the CRT. Examine the printer and restart it by PRINT command.
8	PAPER EMPTY	The printer paper has been exhausted. The ASE stops the printer and continues outputting on the CRT. Supplying papers, restart the printer by PRINT command.
9	PRINTER ALREADY ASSIGNED	The printer has already been assigned. Disassign the former printer and assign the new one.
10	CAN NOT USE THIS MODE	This display mode cannot be selected by the switch on the ASE station. See section 4.2 for more details.
11	SYNTAX ERROR	The command is syntactically incorrect. Correct it.
12	COMMAND NOT FOUND	The command name is incorrect. Correct it.
13	INVALID DATA	The data is incorrect. Correct it.
14	DATA OVERFLOW	The value is greater than 4 bytes. Correct it.
15	SYMBOL TOO LONG	The symbol exceeds 8 characters. Shorten it to within 8.
16	SYMBOL NOT FOUND	The specified symbol is not reginstered. Enter the correct symbol.
17	INVALID ADDRESS	The specified address is incorrect. Correct it.
18	INVALID ADDRESS RANGE	The address range is not specified correctly. Correct it.
19	INVALID PROB OPERAND	PROB operand is incorrect. Correct it.
20	INVALID FLAG REG. OPERAND	Flag register operand is incorrect. Correct it
21	INVALID COMBINATION	Operands are combined incorrectly. Correct them.

No.	Error Message	Description/Solution
22	INVALID RELATIVE EXPRESSION	The relative expressions are combined incorrectly. Correct it.
23	COUNT >1024	The count operand is greater than 1024. Shorten it to within 1024. (This error occurs if it's 0.)
24	CC COMMAND IN COMMAND_CHAIN FILE	COMMAND_CHAIN command is included in the command chain file. As the command chain file cannot be nested, correct the file contents.
25	TOO MANY BREAKPOINTS	The RAM area breakpoints are greater than 255. 256 or more points cannot be specified.
26	BREAKPOINT NOT IN RAM	The specified address is not included in RAM, so a breakpoint cannot be specified in RAM. Use TRACE_BREAK_CONDITION command.
27	TOO MANY SYMBOLS	The registered symbol number is greater than 1000. More than 1000 symbols cannot be specified.
28	INVALID MNEMONIC	The mnemonic specified by the assembler command is invalid. Specify again, correctly.
29	INVALID OPERAND	The operand specified by the assembler command is invalid. Specify again, correctly.
30	SYMBOL IN USE	The symbol is in use with BREAK, STOP, or TRACE_BREAK_CONDITION command. It cannot be eliminated. After cancel it, eliminate it.
31	NO BREAK POINT	No breakpoints have been specified. They cannot be displayed.
32	CANNOT CHANGE DMAC ST REG	The status register for DMAC cannot be changed. Execute the user program when changing.
33	OUT OF LOGICAL ADDRESS RANGE	Since logical address is indifinite, the specified physical address can not be used. Use correct address.
34	CAN NOT SET AT THIS ADDRESS	A breakpoint cannot be specified at $2F (logical address). Use TRACE_BREAK_CONDITION command.
36	CAN NOT USE BUFFER BOX	The type is not the same between BUFFER BOX and ASE program. Check type or version and change so that both are the same.
38	NO ACCESS ADDRESS	The accessed address is not in the specified address range. COVERAGE information is not displayed.
39	VERIFY ERROR	A verify error occurred when comparing with the memory contents. Check the memory attribute and compared data.
40	NO AREA IN SYSTEM MEMORY	There is no free space in the emulation memory. Specify the memory attribute after checking the rest memory.
41	TRACE BUFFER EMPTY	No trace information is in the trace buffer.
42	WRITE TO READ ONLY MEMORY	Data was written into memory defined as "read only" in MAP command.
45	FILE NOT FOUND	The specified file name not found. Specify correct file name.

No.	Error Message	Description/Solution
46	FILE ALREADY EXIST	The specified file name already exists. Choose another name.
47	INVALID FILE NAME	The specified file name is invalid. Specify correctly.
48	INVALID FILE DATA	The file contents are different from the specified or is not the type supported by the ASE. Check the file data.
49	I/O ERROR (nn)	An I/O error occurred between the host system or CRT. Refer to table 10-2 for details with "nn" code.
50	BREAKPOINTS OF THIS AREA DELETED	RAM area break points were deleted in the specified area. (The memory area where data was written.) The processing is continued. (warning message)
52	MAPPING BOUND MUST 4KB	The memory attribute must be set by 4KB unit. It was changed so that the memory bound is at each 4KB. (warning message)
53	OUT OF COVERAGE PAGE	The addresses except the addresses captured by COVERAGE command are included. The range which can be displayed from the specified range will be displayed. (warning message)
55	COMMAND LINE TOO LONG	The specified condition with TRACE_BREAK_CONDITION or STOP command exceeds the size displayed as a former condition. The condition is accepted correctly, but when displaying the former condition, the latter part is not displayed. (warning message)
60	FD NOT READY	READ/WRITE cannot be done because a floppy disk is not in a drive correctly. Check the floppy disk.
61	FD WRITE PROTECT	Data cannot be written because the floppy disk is set write protect. The write protect should be removed.
62	FD NO DATA	The data is not in a floppy disk. Set the correct floppy disk.
63	FD CRC ERROR	CRC error occurs with READ/WRITE of a floppy disk. Format the floppy disk again, or exchange it with new one.
64	FD IS SYSTEM FD	A floppy disk is not used with FLOPPY_LOAD, FLOPPY_SAVE or FLOPPY_VERIFY command since it is a system disk. Use another floppy disk.
65	FD IS NOT SYSTEM FD	A floppy disk to be verified by BACKUP command is not a system disk. Use correct floppy disk.
66	UNMATCH SYSTEM PROGRAM SIZE	The size of the ASE program in a floppy disk to be verified by BACKUP command is different from that of the ASE program in the memory. Use correct floppy disk.
67	UNFORMATTED FD	A floppy disk is not formatted. Use it after formatting with FLOPPY_FORMAT command.
69	FD I/O ERROR (nn)	READ/WRITE processing of a floppy disk causes an error. The floppy disk may have something wrong. Exchange it with another floppy disk or format it again. "nn" means the error reason. Refer to table 10-3.

Table 10-2 Detailed I/O Error Code

No.	Error No.	Description/Solution
1	D1	Parity error occurs with data transfer to the host system. Parity setting of HOST command should be matched to the host system connected.
2	D2	Overrun occurs with data transfer to the host system. BUSY information controlled by the ASE is not realized by the host system. Check the interface written in section 4.3.
3	D3	A framing error occurs with data transfer to the host system. Each information of HOST command should be matched to the host system connected.
4	DC	Time out error occurs with data transfer to the host system. Check the connection with the host system and the operation state of the host system.
5	OA	Tried to prepare a file on SD200 with SAVE command, but the directry area is not enough. Exchange the floppy disk.
6	CD	Tried to prepare a file on SD200 with SAVE command, but the blank area is not enough. Exchange the floppy disk.
7	EB	Tried to input the file data on SD200, but the data is not correct. Check the file data.

Table 10-3 Detailed Floppy Disk Error Code

No.	Error No.	Description/Solution
1	01	Non-existent command is input to floppy disk controller (FDC). Reload the ASE program and activate it again.
2	02	Non-existent floppy disk drive is tried to be used. Reload the ASE program and activate it again.
3	03	Incorrect sector number is tried to be accessed. Reload the ASE program and activate it again.
4	04	FDC is not initialized. Reload the ASE program and activate it again.
5	05	During command processing to FDC, the next command is input again. Reload the ASE program and activate it again.
6	11	A floppy disk is not in a drive. Set a floppy disk.
7	12	The floppy disk has a write protect. The write protect should be removed.
8	21	Data transfer between a floppy disk and the memory is failed. Input the command again.
9	32	The sector to be accessed does not exist. Use it again after formatting.
10	33	"Deleted Data Mark" is detected. Use it again after formatting.
11	34	Time out is detected. Use it again after formattig.
12	41	An error occurs during head moving. Input the a command again.
13	51	The floppy disk is still in BUSY state. Input the command again after pressing a BREAK key.
14	52	FAULT signal is received from FDC. Input the command again after pressing a BREAK key.
15	53	"End of Sector" is detected. Input the command again after pressing a BREAK key.
16	54	FDC malfunctions. Input the command again after pressing a BREAK key.

11. PRECAUTIONS

This chapter provides precautions you have to pay attention when using the ASE. Please read the following carefully.

11.1 Precautions before System Start-up

(1) Confirm that no components on Table 2-1 is missing after unpacking.

(2) Do not put any strength on the chassis such as placing the heavy thing on it.

(3) The ASE has the internal cooling fan on the side chassis of the ASE station and the buffer box. So, please keep the side chassis off the wall or another equipments more than 20cm.

(4) Do not put the ASE into the dusty place.

(5) Do not put the ASE to the hot place such as shiny place or the place near a heater. Also, do not use at the place where the temperature or the humidity changes a lot.

(6) Do not put the ASE into the place where there is too much vibration.

(7) Do not damage by giving the too much impact against or pressure on the ASE.

(8) Please use the ASE power supply when the conditions are met such as the power, the voltage and power supply frequency.

(9) When moving the ASE, be careful not to shock or damage it. Especially, pay particular attention to the parts such as power switch or I/O connectors.

(10) After connecting cables, please check the connecting position again. (Refer to "3. INSTALLATION HARDWARE" for details.) After connecting all cables, power each switch ON. Do not connect I/O cables with power ON.

(11) For MPU IC socket used in the user system, use IC83-64075GG4 (manufactured by Yamaichi Electric Mfg., Ltd.) or equivalent.

(12) When using the ASE system program, use a copy which is prepared to backup with BACKUP command (refer to "9.4 BACKUP") beforehand.

(13) The ASE gets the power supply 54mA max. from user system through user system interface cable (for clock oscillator). Be careful that if the power is not supplied to Vcc pin of MPU connector, the ASE cannot perform the emulation by clock input from the user system.

11.2 Precautions at Emulation

(1) In the following condition, the stack area occupies two bytes in the system. Please provide two bytes more than the stack area used in the user program.

(a) When stop the user program with software break (BREAK command).
(b) When specify hardware break (TRACE_BREAK_CONDITION command) to SLP, HALT instruction and execute this instruction. (At this time, the console displays PC value (the starting address of NMI process servicing routine).)
(c) When execute SLP, HALT instruction with single-step (STEP command) (At this time, PC value is the starting address of NMI process servicing routine.)

(2) During single-step execution, the following signals are not input.

- RESET
- NMI
- INT0, INT1, INT2
- BUSREQ
- DREQ1, DREQ2

Note that the internal interrupt is acceptable.

(3) When using the timer, if you break the user program with single step or breakpoint, the ASE cannot perform the real-time emulation.

(4) If you execute SLP, HALT instruction with single-step command, the ASE cancel SLP, HALT with NMI. So, PC value is the starting address ($0066) of NMI process servicing routine.
Also, if you execute SLEEP, HALT instruction with GO command and press the BREAK key, the PC value is $0066 as well as single-step.

(5) When you break with specifying software break command (BREAK command), if the stack pointer value is not in the RAM area, the ASE displays that it executes RST $30 instruction and stops. (Software break is executed using RST instruction.) Also, PC value is displayed as vectoring address of RST $30 instruction. Note that the ASE displays "BREAK POINT" when the stack pointer value is in the RAM area.

(6) The ASE inputs RESET to the CPU (HD64180) under the following conditions. The ASE displays "RESET IN BY ASE".
(a) The BREAK key is pressed when BUSREQ = Low.
(b) The BREAK key is pressed when RESET = Low.
(c) The BREAK key is pressed when WAIT = Low.
(d) The clock is modified with CLOCK command.
(e) The ASE start-up and termination.

(Note) (a) to (c) occur only when "Low" is sequentially input for more than 80 μS.

(7) In the case of inputting the clock of the MPU (HD64180) from the user system, the ASE is in command input wait state or user program execution state. If the user clock stops (power off, etc.), the ASE displays "USER SYSTEM NOT READY", and returns to the initial state.

(8) If the hardware break condition (TRACE_BREAK_CONDITION command) is met during DMA transfer(burst mode), the ASE does not break immediately. But after DMA transfer finished, the ASE breaks with excuting the first instruction.

(9) During DMA transfer (cycle steal mode), if you break the ASE with the BREAK command (BREAK, TRACE_BREAK_CONDITION command), the DMA stops.(The ASE inserts NMI into the program when stops the user program.) If you executes the GO command again, the ASE makes the DME bit ENABLE just before the user program is exectuted. Therefore, when executing the user program, several bytes of DMA is transferred. (Not saved as a trace information.)

(10) During user program execution under interrupt enable (EI), if the ASE breaks NMI servicing ($0066 up to RETN instruction) (with BREAK, TRACE_BREAK_CONDITION command) and to execute the user program sequentially with GO command again, please specify N option. If you execute the user program with the GO command without N option specified, the ASE executes the NMI interrupt servicing in the condition of IEF1 = 1 and IEF2 = 1. (Usually, IEF1 = 0 and IEF2 = 1 during the NMI interrupt servicing.)

(11) Note that when hardware break (TRACE_BREAK_CONDITION command) is specified to SLP or HALT instruction, and NMI signal is "Low" executing these instructions, the ASE doesn't break the user program.

(12) If $\overline{\text{NMI}}$ signal is input just before GO command execution, NMI interrupt is accepted just after user program has started, and then the ASE executes NMI servicing (starting address $66). The MPU detects NMI by noticing the change from "High" to "Low", but note that the ASE always accepts NMI when the level is "Low" in inputting GO command.
Also, when the ASE breaks in NMI servicing routine and $\overline{\text{NMI}}$ signal is still "Low" in restarting, the ASE restarts from NMI servicing routine again. So, please set $\overline{\text{NMI}}$ signal "High" before restarting.

(13) As the ASE has XTAL oscillator, you can emulate with XTAL in the user system. When using it, please take the following in mind.
① The frequency of XTAL to be used it with the frequency from 8 to 12.2888 MHz. If you want to use it with the frequency less than the above, please input external clock through EXTAL pin.
② The oscillator might not work correctly by XTAL to be used or wiring on the user system. Check that ø clock is always a half of XTAL frequency before the emulation.
③ For precautions of the oscillator and board design, please refer to "HD64180 User's Manual".

(14) Precautions on break during DMA transfer.

① When the ASE breaks during DMA transfer, if you restart the program execution with GO command, DMA transfer also gets restarted. Therefore, restart the user program from the breakpoint address to finish the DMA transfer.

② When DMA is transferred to the write protect memory area by MAP command the ASE forcedly stops the user program. In this case, the program cannot get restarted only by DMA transfer of cycle steal mode. Then reset MPU with CLOCK command. If DMA of burst mode is transferred, the ASE forcedly stops after DMA transfer. Therefore, the program can get restarted.

(15) Please pay attention to the following when activating the ASE interface program with HOST command connected to SD200 CP/M-68K host system.

(a) When inputting "PRINT H (CR)", the printer connected to SD200 should be on line. If not, "I/O ERROR" occurs. Then, except when resetting SD200, LOAD, SAVE, VERIFY, or COMMAND CHAIN command cannot be used.

(b) Without inputting "HOST - (CR)" or "QUIT (CR)" if you power off the ASE the ASE interface program on SD200 is still in operation state. In this case, power on the ASE again, enters "HOST TERMINAL MODE", then press a BREAK key. After that press the control key and "C" key at the same time. In this way, the ASE interface program terminates.

(16) The ASE built-in 3.5-inch floppy disk can be output only one file. To output several files, the ASE requires several floppy disks.

(17) If a parity error occurs when using emulation memory, the ASE displays either of the messages shown in Table 11-1 as a display message of execution state. In this case, there must be something wrong with an emulation memory board installed in the ASE station.
A parity error of an emulation memory is displayed as either "TOUT" or "PARITY ERROR" in the next cycle when CPU accesses the emulation memory after the cycle when a parity error actually occurs. Therefore, if CPU does not access the emulation memory after a parity error occurs, "TOUT" or "PARITY ERROR" will not be displayed.

Table 11-1 : Parity Error Display

NO.	Display
1	** TOUT S = * ADDR = xxxx xxxxx
2	** PARITY ERROR

① Note that "TOUT" will be displayed when $\overline{\text{WAIT}}$ signal is continuously input "Low" for 80μS from user system or a parity error occurs in emulation memory.
The address displayed after "TOUT" is the one where CPU accesses the emulation memory after the cycle when a parity error actually occurs. Therefore, when you try to check the address of a parity error, please check the address where CPU accesses READ address of the emulation memory just before the address displayed after "TOUT", referring to the trace data.

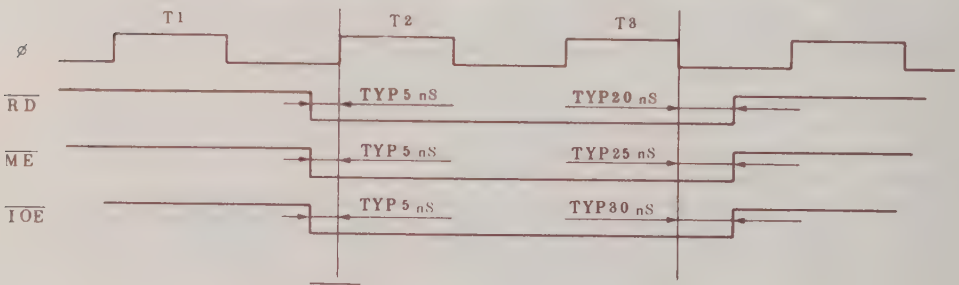
② "PARITY ERROR" will be displayed when CPU accesses the emulation memory in the first op-code fetch cycle after the cycle when a parity error occurs. Therefore, when you try to check the address of a parity error, please check the address where CPU accesses READ address of the emulation memory just before Program Counter (PC) address at the time when the program stops, referring to the trace data.

(18) The signal output from the ASE generates delay through the buffer or cable. Therefore, please adjust the hardware of the user system consideing the above.

(a) The following shows delay from HD64180 to MPU connector in the buffer box.

ø	:	TYP 10 nS
LIR	:	TYP 15 nS
E	:	TYP 12 nS
$\overline{REF}$	:	TYP 10 nS
ADDR	:	TYP 24 nS
DATA	:	TYP 10 nS

(b) The following shows the timing for in the MPU connector.



* The $\overline{WAIT}$ cycle is inserted into $\overline{IOE}$.

12.1 Self Diagnostics

When powered ON (started-up), the ASE performs self diagnostics which checks the internal RAM and the register. The console displays the following and the ASE executes it.

```
ASE MONITOR   Vn.m                                  ------- (1)
COPYRIGHTED BY HITACHI, LTD.  1985.
TESTING
 RAM 01234                                          ------- (2)
SYSTEM MEMORY TEST


ASE SYSTEM MEMORY 01234567  (&xxxKB)                ------- (3)
```

(1) Starting message. "Vn.m" means a version number.

(2) Tests the ASE system RAM (320KB).
Divides the RAM into five parts. Each part corresponds to numbers 0 to 4 and the console displays the number. The numbers represent the following addresses.

0	--------	$10000 to $1FFFF
1	--------	$20000 to $2FFFF
2	--------	$30000 to $3FFFF
3	--------	$40000 to $4BFFF
4	--------	$4C000 to $4FFFF

(3) Tests the emulation memory (option).
Devides the memory into eight parts. Each part corresponds to numbers 0 to 7 and represents 64KB area.
Note that the console displays 0 to 7 only when two emulation memory boards are installed. If only one emulation memory board is installed, the console displays 0 to 3 and if there's no emulation board installed, the console displays only 0.
The number represents the following addresses.

0	--------	$800000 to $80FFFF
1	--------	$810000 to $81FFFF
2	--------	$820000 to $82FFFF
3	--------	$830000 to $83FFFF
4	--------	$840000 to $84FFFF
5	--------	$850000 to $85FFFF
6	--------	$860000 to $86FFFF
7	--------	$870000 to $87FFFF

In the parenthesis, the emulation memory capacity is displayed.

If an error occurs, the display will be as follows.

```
ASE MONITOR  Vn.m
COPYRIGHTED BY HITACHI, LTD.  1985.
TESTING
 RAM 01234
 **RAM ERR ADDR=△△△△△△  W-DATA=xxxxxxxx R-DATA=xxxxxxxx ——— ①
SYSTEM MEMORY TEST

ASE SYSTEM MEMORY 01234567 (&xxxKB)
 **RAM ERR ADDR=△△△△△△  W-DATA=xxxx  R-DATA=xxxx ———————— ②

 **CBR ERROR  W-DATA=xxxx   R-DATA=xxxx ———————————————— ③
 **TCR ERROR  W-DATA=xxxx   R-DATA=xxxx ———————————————— ④
 **BQR ERROR  W-DATA=xxxx   R-DATA=xxxx ———————————————— ⑤
 **RAR ERROR  W-DATA=xxxx   R-DATA=xxxx ———————————————— ⑥
 **TBM ERROR  W-DATA=xxxx   R-DATA=xxxx ———————————————— ⑦
 **PAR11  CLR   ERROR ————————————————————————————————— ⑧
```

① Results from the ASE system RAM test.
The write data and read data differ from each other at $△△△△△△.
② Results from an emulation memory test.
The write data and read data differ from each other at $△△△△△△.
③ The data verify error occurs with READ AFTER WRITE of CBR.
④ The data verify error occurs with READ AFTER WRITE of TCR.
⑤ The data verify error occurs with READ AFTER WRITE of BQR.
⑥ The data verify error occurs with READ AFTER WRITE of RAR.
⑦ The data verify error occurs with READ AFTER WRITE of TBM.
⑧ The eleventh bit of RAR cannot be cleared.
(Note) CBR, TCR, BQR, RAR and TBM are the internal registers of the ASE.

12.2 Troubleshooting

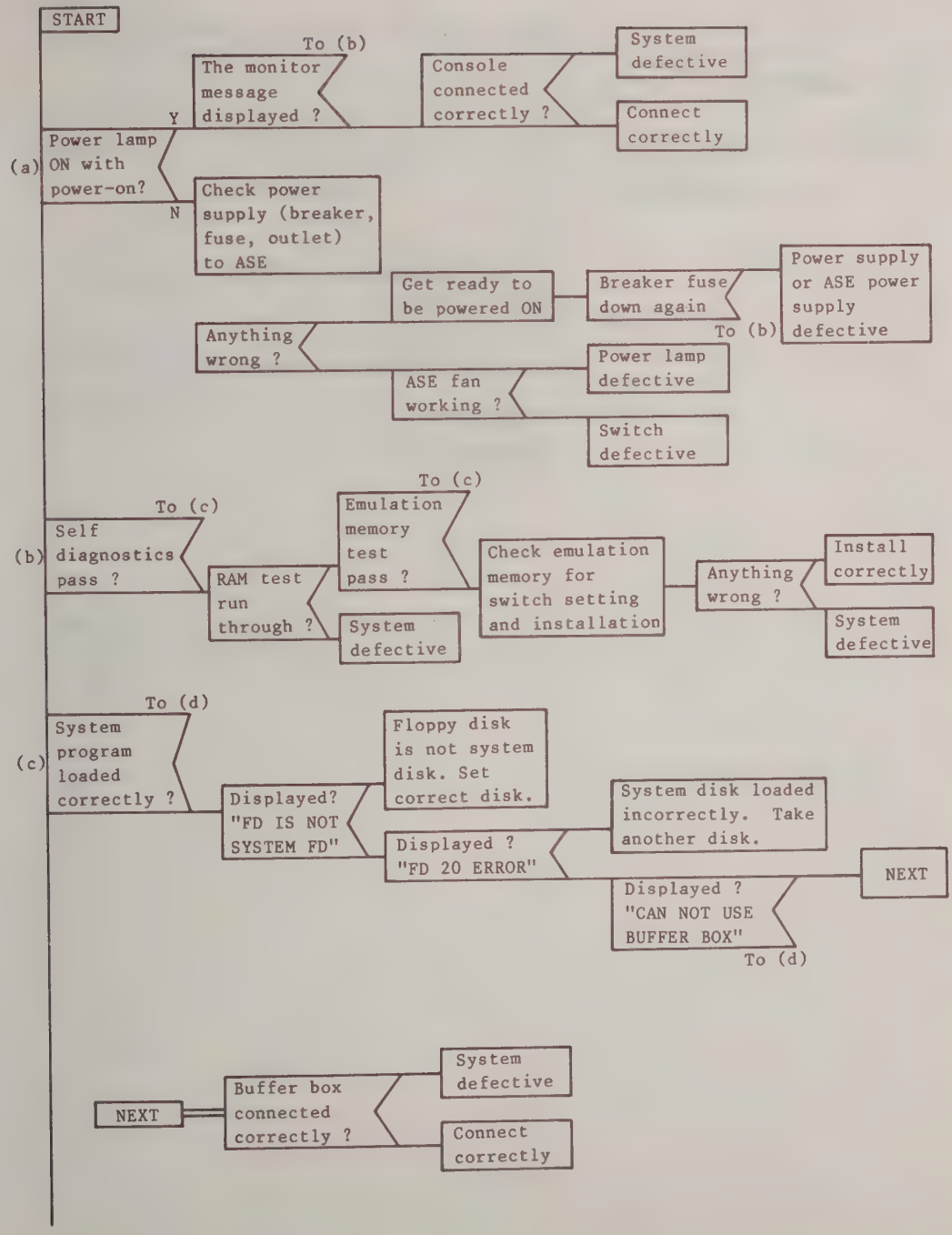
This section aims at reducing the time required for the system to recover from a trouble. It is written within the range the user can handle.

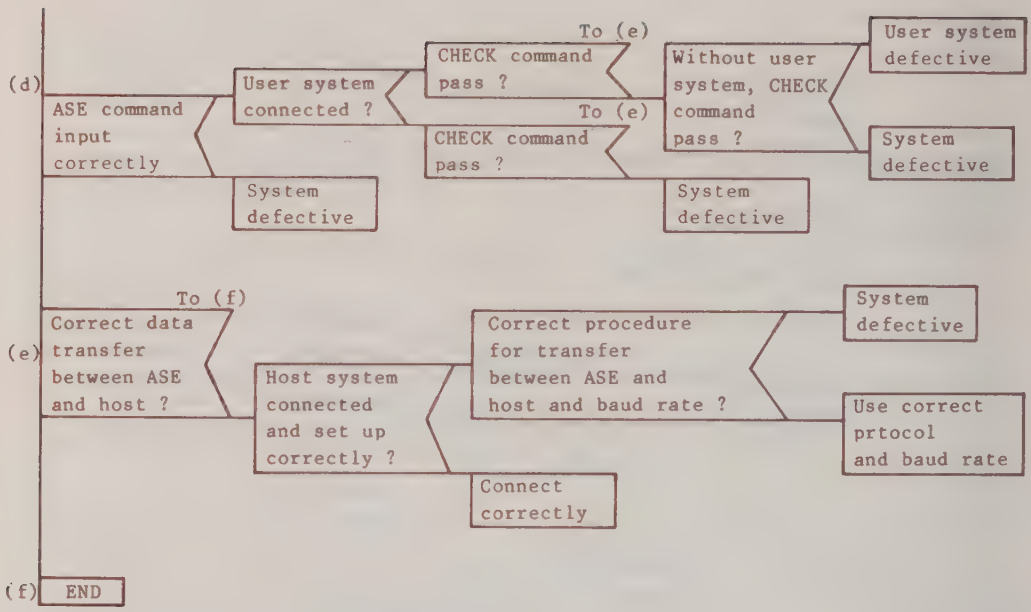
Using the troubleshooting Problem Analysis Diagram (PAD, Fig. 12-1), identify the location where the problem has occurred.

During trace of the diagram;

(a) If you encounter the box meaning "do something", then take the appropriate action and try to operate the system.

(b) If you encounter the box marked "system defective", it means that the ASE station is defective. Inform us of the problem you have discovered. Normally, a trouble will be many factors and cannot always be identified by using PAD. The trouble should be reported as detailedly as possible.


START
To (b)
The monitor message displayed ?
Y
Console connected correctly ?
System defective
Connect correctly
(a) Power lamp ON with power-on?
N
Check power supply (breaker, fuse, outlet) to ASE
Get ready to be powered ON
Breaker fuse down again
Power supply or ASE power supply defective
To (b)
Anything wrong ?
Power lamp defective
ASE fan working ?
Switch defective
To (c)
To (c)
(b) Self diagnostics pass ?
Emulation memory test pass ?
RAM test run through ?
System defective
Check emulation memory for switch setting and installation
Anything wrong ?
Install correctly
System defective
To (d)
(c) System program loaded correctly ?
Displayed? "FD IS NOT SYSTEM FD"
Floppy disk is not system disk. Set correct disk.
Displayed ? "FD 20 ERROR"
System disk loaded incorrectly. Take another disk.
Displayed ? "CAN NOT USE BUFFER BOX"
NEXT
To (d)
NEXT
Buffer box connected correctly ?
System defective
Connect correctly



(Note)

----- This means a question. If the answer is "YES", you take the upper path (Y-side). If "NO", you take the lower one (N-side).

Fig. 12-1 Troubleshooting PAD

Appendix A. The ASE Load Module Format

The ASE uses Motorola S-type format when its host system is SD200 CP/M-68K. Therefore, it is possible to load the object file which CP/M-68K 64180 cross macro assembler prepares. For more details, refer to "CP/M-68K 64180 Cross Macro Assembler User's Manual".

If you connect the host system except SD200 to the ASE, use either of the INTELLEC HEX format or Motorola S-type format.

A.1 describes the Motorola S-type format and A-2 describes the INTELLEC HEX format.

• Motorola S-type record format

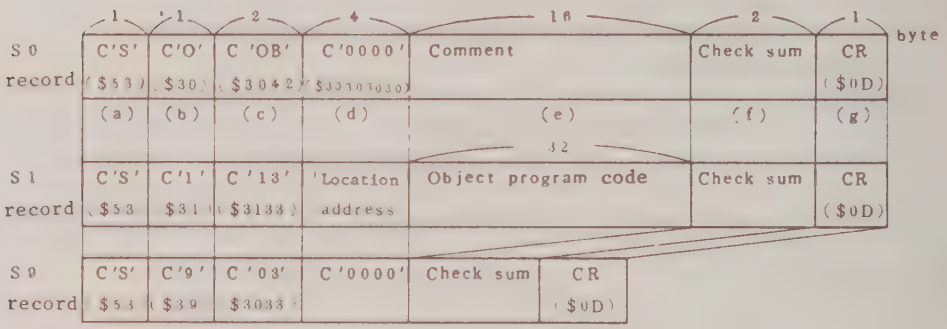


Table A-1 The contents of the Motorola S-type record

No	Items \ Record	S0 record	S1 record	S9 record	Column
1	Record mark	Begins with the 'S' of ASCII character.	←	←	(a)
2	Record type	The code '0' which indicates the record type.	The code '1' which indicates the record type.	The code '9' which indicates the record type.	(b)
3	Byte count	2 digits byte count. Indicated by hexadecimal number, and half the the length from the Location address to the check sum.			(c)
4	Location address	4 digits address. No meaning	4 digits address. The start address when the S1 block is loaded in the memory.	4 digits address No meaning	(d)
5	Data	16 digits comment area	32 digits object code	none	(e)
6	Check sum	A complement of 1 corresponds to the sum total from the byte count to the check sum.			(f)
7	Return	CR ($0D)			(g)

- When N option (SAVE command) is specified to the ASE, in the gaps between the records, LF($0A) and 4 Nulls are output.
- From the byte count to the check sum, each value of hexadecimal number is written converting into ASCII code.
- Reads the termination record up to CR, then terminates.

INTELLEC HEX type record format

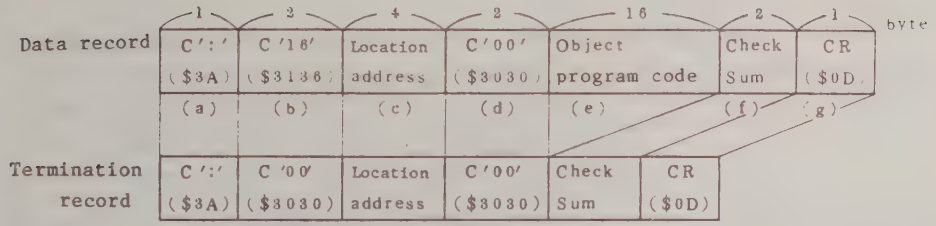


Table A-2 The contents of the INTELLEC HEX type record

	Item \ Record	Data record	Termination record	Column
1	Record mark	':' (colon, $3A) Indicates the record start		(a)
2	Byte count	Indicates the data number in 1 record. Max.16	Indicates the data number in 1 record. In this case, '00'	(b)
3	Location address	Indicates the starting address of the record.	In this case, '0000'.	(c)
4	Record type	'00'	'01'	(d)
5	Data	The code which is stored into the memory.	None	(e)
6	Check sum	A complement of 2 which corresponds to the sum total from the byte count to the check sum.		(f)
7	Return	CR ($0D)		(g)

- Reads the termination record up to CR, then terminates.
- From the byte count to the check sum, each value of hexadecimal number is written converting into ASCII code.
- When N option (SAVE command) is specified to the ASE, in the gaps between the records, LF($0A) and 4 Nulls are output.

Table B-1 ASCII Code

Parity bit				b_8								
				b_7	○	○	○	○	●	●	●	●
				b_6	○	○	●	●	○	○	●	●
				b_5	○	●	○	●	○	●	○	●
b_4	b_3	b_2	b_1	MSD / LSD	0	1	2	3	4	5	6	7
○	○	○	○	0	NUL	DC0	SP	0	@	P	`	p
○	○	○	●	1	SOM	X-ON	!	1	A	Q	a	q
○	○	●	○	2	EOA	TAPE	"	2	B	R	b	r
○	○	●	●	3	EOM	X OFF	#	3	C	S	c	s
○	●	○	○	4	EOT	TAPE	$	4	D	T	d	t
○	●	○	●	5	WRU	ERROR	%	5	E	U	e	u
○	●	●	○	6	RU	SYNC	&	6	F	V	f	v
○	●	●	●	7	BELL	LEM	(APOS)	7	G	W	g	w
●	○	○	○	8	FE_0	CAN	(	8	H	X	h	x
●	○	○	●	9	TAB	S_1	)	9	I	Y	i	y
●	○	●	○	A	LF	EOF	*	:	J	Z	j	z
●	○	●	●	B	VT	ESC	+	;	K	[	k	{
●	●	○	○	C	FF	S_4	(COMMA)	<	L	\	l	|
●	●	○	●	D	CR	S_5	-	=	M	]	m	}
●	●	●	○	E	SO	S_6	.	>	N	^	n	~
●	●	●	●	F	SI	S_7	/	?	O	_	o	RUB OUT

Hexadecimal-Decimal Number Conversion Table (1)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	0010	0011	0012	0013	0014	0015
01	0016	0017	0018	0019	0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	0030	0031
02	0032	0033	0034	0035	0036	0037	0038	0039	0040	0041	0042	0043	0044	0045	0046	0047
03	0048	0049	0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	0060	0061	0062	0063
04	0064	0065	0066	0067	0068	0069	0070	0071	0072	0073	0074	0075	0076	0077	0078	0079
05	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	0090	0091	0092	0093	0094	0095
06	0096	0097	0098	0099	0100	0101	0102	0103	0104	0105	0106	0107	0108	0109	0110	0111
07	0112	0113	0114	0115	0116	0117	0118	0119	0120	0121	0122	0123	0124	0125	0126	0127
08	0128	0129	0130	0131	0132	0133	0134	0135	0136	0137	0138	0139	0140	0141	0142	0143
09	0144	0145	0146	0147	0148	0149	0150	0151	0152	0153	0154	0155	0156	0157	0158	0159
0A	0160	0161	0162	0163	0164	0165	0166	0167	0168	0169	0170	0171	0172	0173	0174	0175
0B	0176	0177	0178	0179	0180	0181	0182	0183	0184	0185	0186	0187	0188	0189	0190	0191
0C	0192	0193	0194	0195	0196	0197	0198	0199	0200	0201	0202	0203	0204	0205	0206	0207
0D	0208	0209	0210	0211	0212	0213	0214	0215	0216	0217	0218	0219	0220	0221	0222	0223
0E	0224	0225	0226	0227	0228	0229	0230	0231	0232	0233	0234	0235	0236	0237	0238	0239
0F	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255
10	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271
11	0272	0273	0274	0275	0276	0277	0278	0279	0280	0281	0282	0283	0284	0285	0286	0287
12	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	0301	0302	0303
13	0304	0305	0306	0307	0308	0309	0310	0311	0312	0313	0314	0315	0316	0317	0318	0319
14	0320	0321	0322	0323	0324	0325	0326	0327	0328	0329	0330	0331	0332	0333	0334	0335
15	0336	0337	0338	0339	0340	0341	0342	0343	0344	0345	0346	0347	0348	0349	0350	0351
16	0352	0353	0354	0355	0356	0357	0358	0359	0360	0361	0362	0363	0364	0365	0366	0367
17	0368	0369	0370	0371	0372	0373	0374	0375	0376	0377	0378	0379	0380	0381	0382	0383
18	0384	0385	0386	0387	0388	0389	0390	0391	0392	0393	0394	0395	0396	0397	0398	0399
19	0400	0401	0402	0403	0404	0405	0406	0407	0408	0409	0410	0411	0412	0413	0414	0415
1A	0416	0417	0418	0419	0420	0421	0422	0423	0424	0425	0426	0427	0428	0429	0430	0431
1B	0432	0433	0434	0435	0436	0437	0438	0439	0440	0441	0442	0443	0444	0445	0446	0447
1C	0448	0449	0450	0451	0452	0453	0454	0455	0456	0457	0458	0459	0460	0461	0462	0463
1D	0464	0465	0466	0467	0468	0469	0470	0471	0472	0473	0474	0475	0476	0477	0478	0479
1E	0480	0481	0482	0483	0484	0485	0486	0487	0488	0489	0490	0491	0492	0493	0494	0495
1F	0496	0497	0498	0499	0500	0501	0502	0503	0504	0505	0506	0507	0508	0509	0510	0511
20	0512	0513	0514	0515	0516	0517	0518	0519	0520	0521	0522	0523	0524	0525	0526	0527
21	0528	0529	0530	0531	0532	0533	0534	0535	0536	0537	0538	0539	0540	0541	0542	0543
22	0544	0545	0546	0547	0548	0549	0550	0551	0552	0553	0554	0555	0556	0557	0558	0559
23	0560	0561	0562	0563	0564	0565	0566	0567	0568	0569	0570	0571	0572	0573	0574	0575
24	0576	0577	0578	0579	0580	0581	0582	0583	0584	0585	0586	0587	0588	0589	0590	0591
25	0592	0593	0594	0595	0596	0597	0598	0599	0600	0601	0602	0603	0604	0605	0606	0607
26	0608	0609	0610	0611	0612	0613	0614	0615	0616	0617	0618	0619	0620	0621	0622	0623
27	0624	0625	0626	0627	0628	0629	0630	0631	0632	0633	0634	0635	0636	0637	0638	0639
28	0640	0641	0642	0643	0644	0645	0646	0647	0648	0649	0650	0651	0652	0653	0654	0655
29	0656	0657	0658	0659	0660	0661	0662	0663	0664	0665	0666	0667	0668	0669	0670	0671
2A	0672	0673	0674	0675	0676	0677	0678	0679	0680	0681	0682	0683	0684	0685	0686	0687
2B	0688	0689	0690	0691	0692	0693	0694	0695	0696	0697	0698	0699	0700	0701	0702	0703
2C	0704	0705	0706	0707	0708	0709	0710	0711	0712	0713	0714	0715	0716	0717	0718	0719
2D	0720	0721	0722	0723	0724	0725	0726	0727	0728	0729	0730	0731	0732	0733	0734	0735
2E	0736	0737	0738	0739	0740	0741	0742	0743	0744	0745	0746	0747	0748	0749	0750	0751
2F	0752	0753	0754	0755	0756	0757	0758	0759	0760	0761	0762	0763	0764	0765	0766	0767
30	0768	0769	0770	0771	0772	0773	0774	0775	0776	0777	0778	0779	0780	0781	0782	0783
31	0784	0785	0786	0787	0788	0789	0790	0791	0792	0793	0794	0795	0796	0797	0798	0799
32	0800	0801	0802	0803	0804	0805	0806	0807	0808	0809	0810	0811	0812	0813	0814	0815
33	0816	0817	0818	0819	0820	0821	0822	0823	0824	0825	0826	0827	0828	0829	0830	0831
34	0832	0833	0834	0835	0836	0837	0838	0839	0840	0841	0842	0843	0844	0845	0846	0847
35	0848	0849	0850	0851	0852	0853	0854	0855	0856	0857	0858	0859	0860	0861	0862	0863
36	0864	0865	0866	0867	0868	0869	0870	0871	0872	0873	0874	0875	0876	0877	0878	0879
37	0880	0881	0882	0883	0884	0885	0886	0887	0888	0889	0890	0891	0892	0893	0894	0895

Hexadecimal-Decimal Number Conversion Table (2)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
38	0896	0897	0898	0899	0900	0901	0902	0903	0904	0905	0906	0907	0908	0909	0910	0911
39	0912	0913	0914	0915	0916	0917	0918	0919	0920	0921	0922	0923	0924	0925	0926	0927
3A	0928	0929	0930	0931	0932	0933	0934	0935	0936	0937	0938	0939	0940	0941	0942	0943
3B	0944	0945	0946	0947	0948	0949	0950	0951	0952	0953	0954	0955	0956	0957	0958	0959
3C	0960	0961	0962	0963	0964	0965	0966	0967	0968	0969	0970	0971	0972	0973	0974	0975
3D	0976	0977	0978	0979	0980	0981	0982	0983	0984	0985	0986	0987	0988	0989	0990	0991
3E	0992	0993	0994	0995	0996	0997	0998	0999	1000	1001	1002	1003	1004	1005	1006	1007
3F	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023
40	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039
41	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055
42	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071
43	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087
44	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103
45	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119
46	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135
47	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151
48	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167
49	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183
4A	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199
4B	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215
4C	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231
4D	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247
4E	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263
4F	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279
50	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295
51	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311
52	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327
53	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343
54	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359
55	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375
56	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391
57	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407
58	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423
59	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439
5A	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455
5B	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471
5C	1472	1473	1474	1475	1476	1477	1478	1479	1480	1481	1482	1483	1484	1485	1486	1487
5D	1488	1489	1490	1491	1492	1493	1494	1495	1496	1497	1498	1499	1500	1501	1502	1503
5E	1504	1505	1506	1507	1508	1509	1510	1511	1512	1513	1514	1515	1516	1517	1518	1519
5F	1520	1521	1522	1523	1524	1525	1526	1527	1528	1529	1530	1531	1532	1533	1534	1535
60	1536	1537	1538	1539	1540	1541	1542	1543	1544	1545	1546	1547	1548	1549	1550	1551
61	1552	1553	1554	1555	1556	1557	1558	1559	1560	1561	1562	1563	1564	1565	1566	1567
62	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578	1579	1580	1581	1582	1583
63	1584	1585	1586	1587	1588	1589	1590	1591	1592	1593	1594	1595	1596	1597	1598	1599
64	1600	1601	1602	1603	1604	1605	1606	1607	1608	1609	1610	1611	1612	1613	1614	1615
65	1616	1617	1618	1619	1620	1621	1622	1623	1624	1625	1626	1627	1628	1629	1630	1631
66	1632	1633	1634	1635	1636	1637	1638	1639	1640	1641	1642	1643	1644	1645	1646	1647
67	1648	1649	1650	1651	1652	1653	1654	1655	1656	1657	1658	1659	1660	1661	1662	1663
68	1664	1665	1666	1667	1668	1669	1670	1671	1672	1673	1674	1675	1676	1677	1678	1679
69	1680	1681	1682	1683	1684	1685	1686	1687	1688	1689	1690	1691	1692	1693	1694	1695
6A	1696	1697	1698	1699	1700	1701	1702	1703	1704	1705	1706	1707	1708	1709	1710	1711
6B	1712	1713	1714	1715	1716	1717	1718	1719	1720	1721	1722	1723	1724	1725	1726	1727
6C	1728	1729	1730	1731	1732	1733	1734	1735	1736	1737	1738	1739	1740	1741	1742	1743
6D	1744	1745	1746	1747	1748	1749	1750	1751	1752	1753	1754	1755	1756	1757	1758	1759
6E	1760	1761	1762	1763	1764	1765	1766	1767	1768	1769	1770	1771	1772	1773	1774	1775
6F	1776	1777	1778	1779	1780	1781	1782	1783	1784	1785	1786	1787	1788	1789	1790	1791

Hexadecimal-Decimal Number Conversion Table (3)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
70	1792	1793	1794	1795	1796	1797	1798	1799	1800	1801	1802	1803	1804	1805	1806	1807
71	1808	1809	1810	1811	1812	1813	1814	1815	1816	1817	1818	1819	1820	1821	1822	1823
72	1824	1825	1826	1827	1828	1829	1830	1831	1832	1833	1834	1835	1836	1837	1838	1839
73	1840	1841	1842	1843	1844	1845	1846	1847	1848	1849	1850	1851	1852	1853	1854	1855
74	1856	1857	1858	1859	1860	1861	1862	1863	1864	1865	1866	1867	1868	1869	1870	1871
75	1872	1873	1874	1875	1876	1877	1878	1879	1880	1881	1882	1883	1884	1885	1886	1887
76	1888	1889	1890	1891	1892	1893	1894	1895	1896	1897	1898	1899	1900	1901	1902	1903
77	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919
78	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935
79	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951
7A	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967
7B	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983
7C	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
7D	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
7E	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
7F	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047
80	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063
81	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079
82	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095
83	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111
84	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
85	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143
86	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159
87	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175
88	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191
89	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
8A	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223
8B	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239
8C	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255
8D	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271
8E	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
8F	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303
90	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319
91	2320	2321	2322	2323	2324	2325	2326	2327	2328	2329	2330	2331	2332	2333	2334	2335
92	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351
93	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
94	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383
95	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399
96	2400	2401	2402	2403	2404	2405	2406	2407	2408	2409	2410	2411	2412	2413	2414	2415
97	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431
98	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
99	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463
9A	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479
9B	2480	2481	2482	2483	2484	2485	2486	2487	2488	2489	2490	2491	2492	2493	2494	2495
9C	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511
9D	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
9E	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543
9F	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559
A0	2560	2561	2562	2563	2564	2565	2566	2567	2568	2569	2570	2571	2572	2573	2574	2575
A1	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591
A2	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
A3	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623
A4	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639
A5	2640	2641	2642	2643	2644	2645	2646	2647	2648	2649	2650	2651	2652	2653	2654	2655
A6	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671
A7	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687

Hexadecimal-Decimal Number Conversion Table (4)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
A8	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703
A9	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719
AA	2720	2721	2722	2723	2724	2725	2726	2727	2728	2729	2730	2731	2732	2733	2734	2735
AB	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751
AC	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
AD	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783
AE	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799
AF	2800	2801	2802	2803	2804	2805	2806	2807	2808	2809	2810	2811	2812	2813	2814	2815
B0	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831
B1	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
B2	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863
B3	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879
B4	2880	2881	2882	2883	2884	2885	2886	2887	2888	2889	2890	2891	2892	2893	2894	2895
B5	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911
B6	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
B7	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943
B8	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959
B9	2960	2961	2962	2963	2964	2965	2966	2967	2968	2969	2970	2971	2972	2973	2974	2975
BA	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991
BB	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
BC	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023
BD	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039
BE	3040	3041	3042	3043	3044	3045	3046	3047	3048	3049	3050	3051	3052	3053	3054	3055
BF	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071
C0	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
C1	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103
C2	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119
C3	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135
C4	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
C5	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
C6	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183
C7	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199
C8	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215
C9	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
CA	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
CB	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263
CC	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279
CD	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295
CE	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
CF	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
D0	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343
D1	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359
D2	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375
D3	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
D4	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
D5	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423
D6	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439
D7	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455
D8	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
D9	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
DA	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503
DB	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519
DC	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535
DD	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
DE	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
DF	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583

Hexadecimal-Decimal Number Conversion Table (5)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
E0	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599
E1	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615
E2	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
E3	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
E4	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663
E5	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679
E6	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695
E7	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
E8	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
E9	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743
EA	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759
EB	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775
EC	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
ED	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807
EE	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823
EF	3824	3825	3826	3827	3828	3829	3830	3831	3832	3833	3834	3835	3836	3837	3838	3839
F0	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855
F1	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
F2	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887
F3	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903
F4	3904	3905	3906	3907	3908	3909	3910	3911	3912	3913	3914	3915	3916	3917	3918	3919
F5	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935
F6	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
F7	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967
F8	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983
F9	3984	3985	3986	3987	3988	3989	3990	3991	3992	3993	3994	3995	3996	3997	3998	3999
FA	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015
FB	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
FC	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047
FD	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063
FE	4064	4065	4066	4067	4068	4069	4070	4071	4072	4073	4074	4075	4076	4077	4078	4079
FF	4080	4081	4082	4083	4084	4085	4086	4087	4088	4089	4090	4091	4092	4093	4094	4095

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 949 953 421 312	49	0.000 000 000 000 001 776 356 839 400 250 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0.000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0.000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0.000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0.000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0.000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0.000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5
72 057 594 037 927 936	56	0.000 000 000 000 000 013 877 787 807 814 456 755 295 395 851 135 253 906 25
144 115 188 075 855 872	57	0.000 000 000 000 000 006 938 893 903 907 228 377 647 697 925 567 626 953 125
288 230 376 151 711 744	58	0.000 000 000 000 000 003 469 446 951 953 614 188 823 848 962 783 813 476 562 5
576 460 752 303 423 488	59	0.000 000 000 000 000 001 734 723 475 976 807 094 411 924 481 391 906 738 281 25
1 152 921 504 606 846 976	60	0.000 000 000 000 000 000 867 361 737 988 403 547 205 962 240 695 953 369 140 625
2 305 843 009 213 693 952	61	0.000 000 000 000 000 000 433 680 868 994 201 773 602 981 120 347 976 684 570 312 5
4 611 686 018 427 387 904	62	0.000 000 000 000 000 000 216 840 434 497 100 886 801 490 560 173 988 342 285 156 25
9 223 372 036 854 775 808	63	0.000 000 000 000 000 000 108 420 217 248 550 443 400 745 280 086 994 171 142 578 125
18 446 744 073 709 551 616	64	0.000 000 000 000 000 000 054 210 108 624 275 221 700 372 640 043 497 085 571 289 062 5
36 893 488 147 419 103 232	65	0.000 000 000 000 000 000 027 105 054 312 137 610 850 186 320 021 748 542 785 644 531 25
73 786 976 294 838 206 464	66	0.000 000 000 000 000 000 013 552 527 156 068 805 425 093 160 010 874 271 392 822 265 625
147 573 952 589 676 412 928	67	0.000 000 000 000 000 000 006 776 263 578 034 402 712 546 580 005 437 135 696 411 132 812 5
295 147 905 179 352 825 856	68	0.000 000 000 000 000 000 003 388 131 789 017 201 356 273 290 002 718 567 848 205 566 406 25
590 295 810 358 705 651 712	69	0.000 000 000 000 000 000 001 694 065 894 508 600 678 136 645 001 359 283 924 102 783 203 125
180 591 620 717 411 303 424	70	0.000 000 000 000 000 000 000 847 032 947 254 300 339 068 322 500 679 641 962 051 391 601 562 5
361 183 241 434 822 606 848	71	0.000 000 000 000 000 000 000 423 516 473 627 150 169 534 161 250 339 820 981 025 695 800 781 2
722 366 482 869 645 213 696	72	0.000 000 000 000 000 000 000 211 758 236 813 575 084 767 080 625 169 910 490 512 847 900 390 6

HITACHI AMERICA, LTD.
SEMICONDUCTOR AND IC DIVISION

HEADQUARTERS

Hitachi, Ltd.
New Marunouchi Bldg., 5-1,
Marunouchi 1-chome
Chiyoda-ku, Tokyo 100, Japan
Tel: Tokyo (03) 212-1111
Telex: J22395, J22432, J24491,
J26375 (HITACHY)
Cable: HITACHY TOKYO

U.S. SALES OFFICE

Hitachi America, Ltd.
Semiconductor and IC Division
2210 O'Toole Avenue
San Jose, CA 95131
Tel: 408-942-1500
Telex: 17-1581
Twx: 910-338-2103
Fax: 408-942-8225
Fax: 408-942-8880

REGIONAL OFFICES

NORTHEAST REGION

Hitachi America, Ltd.
5 Burlington Woods Drive
Burlington, MA 01803
617/229-2150

SOUTH CENTRAL REGION

Hitachi America, Ltd.
Two Lincoln Centre, Suite 865
5420 LBJ Freeway
Dallas, TX 75240
214/991-4510

NORTH CENTRAL REGION

Hitachi America, Ltd.
500 Park Blvd., Suite 415
Itasca, IL 60143
312/773-4864

NORTHWEST REGION

Hitachi America, Ltd.
2099 Gateway Place, Suite 550
San Jose, CA 95110
408/277-0712

SOUTHWEST REGION

Hitachi America, Ltd.
21600 Oxnard St., Suite 600
Woodland Hills, CA 91367
818/704-6500

SOUTHEAST REGION

Hitachi America, Ltd.
4901 N.W. 17th Way, Suite 302
Fort Lauderdale, FL 33309
305/491-6154

DISTRICT OFFICES

- Hitachi America, Ltd.
  1700 Galloping Hill Rd.
  Kenilworth, NJ 07033
  201/245-6400
- Hitachi America, Ltd.
  3500 W. 80th Street, Suite 660
  Bloomington, MN 55431
  612/831-0408
- Hitachi America, Ltd.
  80 Washington St., Suite 302
  Poughkeepsie, NY 12601
  914/485-3400
- Hitachi America, Ltd.
  6 Parklane Blvd., #558
  Dearborn, MI 48126
  313/271-4410
- Hitachi America, Ltd.
  6161 Savoy Dr., Suite 850
  Houston, TX 77036
  713/974-0534
- Hitachi America, Ltd.
  5775 Peachtree-Dunwoody Rd.
  Suite 270C
  Atlanta, GA 30342
  404/843-3445
- Hitachi America, Ltd.
  18300 Von Karman Avenue, Suite 730
  Irvine, CA 92715
  714/553-8500
- Hitachi America, Ltd.
  10300 S.W. Greenburg Rd., Suite 480
  Portland, OR 97223
  503/245-1825
- Hitachi (Canadian) Ltd.
  2625 Queensview Dr.
  Ottawa, Ontario, Canada K2A 3Y4
  613/596-2777

We make things possible

Hitachi America, Ltd.
Semiconductor and IC Division
2210 O'Toole Avenue, San Jose, CA 95131
1-408-942-1500

5M

Printed in U.S.A.